UNIVERSIDAD POLITÉCNICA SALESIANA

SEDE QUITO

CARRERA: INGENIERÍA DE SISTEMAS

Trabajo de titulación previo a la obtención del título de: INGENIERO DE SISTEMAS E INGENIERA DE SISTEMAS.

TEMA:
ANÁLISIS, DISEÑO, DESARROLLO E INTEGRACIÓN DE LOS MÓDULOS DE GESTIÓN ACADÉMICA DE CARRERAS Y DISPONIBILIDAD DE DOCENTES, DEL SISTEMA UPS SCHEDULE PARA LA UNIVERSIDAD POLITÉCNICA SALESIANA SEDE QUITO

AUTORES:
SANDRA YADIRA SANTILLÁN PROAÑO
RAÚL ALEJANDRO TORRES FLORES

DIRECTOR:
RODRIGO EFRAÍN TUFIÑO CÁRDENAS

Quito, diciembre del 2014

DECLARATORIA DE RESPONSABILIDAD Y AUTORIZACIÓN DEL USO DEL TRABAJO DE TITULACIÓN

Nosotros, autorizamos a la Universidad Politécnica Salesiana la publicación total o parcial de este trabajo de titulación y su reproducción sin fines de lucro.

Además, declaramos que los conceptos y análisis desarrollados y las conclusiones del presente trabajo son de exclusiva responsabilidad de los autores.

Quito, diciembre del 2014

Sandra Yadira Santillán Proaño
CC: 171735054-8

Raúl Alejandro Torres Flores
CC: 171944074-3

ÍNDICE

INTRODUCCIÓN **1**

CAPÍTULO 1 ..... **6**

ANÁLISIS Y DISEÑO **6**

1.1 Análisis de viabilidad 6

1.1.1 Viabilidad técnica. 6

1.1.2 Viabilidad económica. 7

1.1.2.1 Análisis de factibilidad TIR/VAN. 7

1.1.3 Viabilidad operacional. 9

1.1.4 Resumen. 10

1.2 Análisis de requerimientos 10

1.2.1 Alcance 10

1.2.2 Definiciones, acrónimos y abreviaturas. 11

1.2.3 Visión general. 11

1.2.4 Descripción general. 11

1.2.4.1 Perspectivas del producto. 11

1.2.4.2 Funciones del producto. 12

1.2.4.3 Características del usuario. 12

1.2.4.4 Restricciones de diseño. 12

1.2.5 Requerimientos específicos. 13

1.2.5.1 Interfaces de usuario 13

1.2.5.2 Interfaces de hardware 13

1.2.5.3 Interfaces de software. 13

1.2.5.4 Interfaces de comunicaciones. 13

1.2.6 Requerimientos funcionales. 13

1.2.6.1 Administrador del sistema. 14

1.2.6.2 Docente. 15

1.2.7 Requisitos de rendimiento. 16

1.3 Diseño del sistema 16

1.3.1 Diagramas de casos de uso. 17

1.3.1.1 Gestión de carreras. 17

1.3.1.2 Gestión de materias. 18

1.3.1.3 Gestión de niveles. 19

1.3.1.4 Gestión de mallas. 20

1.3.2 Interfaces de usuario. 20

1.3.3 Mapa de navegación. 22

1.3.4 Diagrama de bloques general. 24

1.3.5 Base de datos. 25

1.3.5.1 Diagrama de base de datos. 26

1.3.5.2 Diccionario de datos. 28

CAPÍTULO 2 32

CODIFICACIÓN Y PRUEBAS 32

2.1 Lenguajes de programación, almacenamiento y herramientas 32

2.1.1 Java. 32

2.1.1.1 Primefaces. 32

2.1.1.2 Java Server Faces (JSF). 33

2.1.2 NetBeans 7.1.2 34
2.1.2.1 Glassfish. 34
2.1.3 PostgreSQL. 35
2.2 Codificación 35
2.2.1 Arquitectura del sistema. 36
2.2.1.1 Diagrama de paquetes. 37
2.2.1.2 Framework X-PRIME V. 1.0 38
2.2.2 Paquete: paq_persistencia. 38
2.2.2.1 Clase cla_conexion. 38
2.2.3 Paquete: paq_recurso_primefaces. 39
2.2.3.1 Clase pf_tabla. 39
2.2.3.2 Clase pf_menu. 41
2.2.3.3 Clase pf_arbol. 42
2.2.4 Paquete: paq_gestion_academica. 42
2.2.4.1 Gestión de carreras. 42
2.2.4.2 Gestión de materias. 44
2.2.4.3 Gestión de mallas. 46
2.2.4.4 Gestión de niveles. 47
2.2.4.5 Gestión de materias paracadémicas. 48
2.2.4.6 Disponibilidad de docentes. 49
2.2.4.7 Reportes. 51
2.3 Pruebas 53

2.3.1 Pruebas contra requerimientos. 53

2.3.1.1 Pruebas funcionales. 53

2.3.2 Pruebas de rendimiento. 57

CAPÍTULO 3 64

INTEGRACIÓN 64

CONCLUSIONES 67

RECOMENDACIONES 68

LISTA DE REFERENCIAS 69

ÍNDICE DE TABLAS

Tabla 1. Detalle de los lenguajes o herramientas para el desarrollo del proyecto 6
Tabla 2. Características del computador a usarse 6
Tabla 3. Detalles de la viabilidad económica 7
Tabla 4. Desarrollo TIR/VAN 8
Tabla 5. Análisis del periodo de repago 9
Tabla 6. Análisis costo / beneficio 9
Tabla 7. Requerimiento funcional 1.1: Gestión de carreras 14
Tabla 8. Requerimiento funcional 1.2: Gestión de niveles 14
Tabla 9. Requerimiento funcional 1.3: Gestión de materias 14
Tabla 10. Requerimiento funcional 1.4: Gestión de grupos 14
Tabla 11. Requerimiento funcional 1.5: Gestión de mallas 15
Tabla 12. Requerimiento funcional 1.6: Reporte de carreras 15
Tabla 13. Requerimiento funcional 1.7: Reporte de materias 15
Tabla 14. Requerimiento funcional 2.1: Disponibilidad del docente 15
Tabla 15. Requerimiento funcional 2.2: Modificación en la DID 16
Tabla 16. Requerimiento funcional 2.3: Reporte de disponibilidad del docente 16
Tabla 17. Caso de uso: Gestión de carreras 17
Tabla 18. Caso de uso: Gestión de materias 18
Tabla 19. Caso de uso: Gestión de niveles 19
Tabla 20. Caso de uso: Gestión de mallas 20
Tabla 21. Descripción de las tablas utilizadas 26
Tabla 22. Diccionario de datos – tbl_opcion 28
Tabla 23. Diccionario de datos – tbl_usuario 28
Tabla 24. Diccionario de datos – tbl_perfil 29
Tabla 25. Diccionario de datos – tbl_nivel 29
Tabla 26. Diccionario de datos – tbl_carrera 29
Tabla 27. Diccionario de datos – tbl_dia 29
Tabla 28. Diccionario de datos – tbl_materia 30
Tabla 29. Diccionario de datos – tbl_hora 30

Tabla 30. Diccionario de datos – tbl_estructura 30
Tabla 31. Diccionario de datos – tbl_disponibilidad 31
Tabla 32. Prueba funcional – Administrador 53
Tabla 32. Prueba funcional – Administrador 54
Tabla 32. Prueba funcional – Administrador 55
Tabla 32. Prueba funcional – Administrador 56
Tabla 33. Prueba funcional - Docente 57
Tabla 34. Prueba de rendimiento-funcionalidad de pantallas de GAD y DID..... 58
Tabla 34. Prueba de rendimiento-funcionalidad de pantallas de GAD y DID..... 59

ÍNDICE DE FIGURAS

Figura 1. Diagrama caso de uso – Gestión de carreras 17
Figura 2. Diagrama caso de uso – Gestión de materias 18
Figura 4. Diagrama caso de uso – Gestión de mallas 20
Figura 5. GUI – Administración académica 21
Figura 6. GUI – Administración académica - reportes 22
Figura 8. Mapa navegacional - Docente 24
Figura 9. Diagrama de bloques – Gestión académica de carreras 25
Figura 10. Diagrama de bloques – Disponibilidad de docentes 25
Figura 11. Diagrama Conceptual de la base de datos 27
Figura 12. Diagrama de despliegue 36
Figura 13. Diagrama de paquetes 37
Figura 14. Administración de carreras 44
Figura 15. Administración de materias 46
Figura 16. Administración de mallas 47
Figura 17. Administración de niveles 48
Figura 18. Administración de materias paracadémicas 48
Figura 19. Cuadrícula - días de la semana 50
Figura 20. Cuadrícula – horas de la semana 51
Figura 21. Selección de reporte 52
Figura 22. Tipo de formato de reporte 52
Figura 23. Reporte en formato PDF 52
Figura 24. Tiempo de carga (ms) en pantallas de administración 59
Figura 25. Tiempo en transacciones de AC 60
Figura 26. Tiempo en transacciones de administración de materias 61
Figura 27. Tiempo en transacciones de administración de sedes 61
Figura 28. Tiempo en transacciones de administración de niveles 62
Figura 29. Tiempo en transacciones de administración de materias paracadémicas 62
Figura 30. Tiempo en transacciones de administración de mallas 63
Figura 31. Tiempo en transacciones al generar reporte de materias 63

RESUMEN

La Universidad Politécnica Salesiana, cada semestre requiere diseñar e implementar los horarios que permitirán la organización de las carreras y docentes; estos horarios son diseñados de forma manual por los directores de carrera, los mismos que deben considerar los cruces de horas, en las materias y docentes, así como también considerar la disponibilidad del docente para tareas de asesoramiento académico e investigación. Esto genera consumo de recursos económicos, físicos, tiempo e incluso recurso humano, dificultándose así la estructura de la información para la generación de los horarios.

Al realizar un diseño manual de los horarios y presentarse cambios o actualizaciones en la malla académica, se vuelve compleja la reorganización manual del horario, haciendo que este proceso no sea el más óptimo y eficiente, requiriendo mayores recursos, demorando aún más el diseño del horario.

Se propone el desarrollo de un sistema dividido en módulos que optimice el proceso de generación de horarios. Dos de éstos módulos se especifican en este proyecto de titulación: Gestión académica de carreras y disponibilidad de docentes, creados y desarrollados bajo los parámetros establecidos por todos los grupos de trabajo en cuanto a tecnología y herramientas de desarrollo.

En el presente trabajo se definen el diseño y las características de los módulos: su interacción con los usuarios, la tarea que cumplen dentro del sistema UPS Schedule, la integración realizada para su correlación con otros módulos, a fin de contribuir y complementar el funcionamiento del sistema completo.

ABSTRACT

The Universidad Politécnica Salesiana, each semester requires designing and implementing schedules that will allow the organization of careers and teachers; these schedules are designed manually by the career directors, the schedules must consider time crossing, in subjects and teachers, as well as the availability of teachers for academic advising tasks and research. This generates consumption of economic, physical, time and even human resources, making more difficult the information structure for the generation of schedules.

When performing a manual design of schedules and changes or updates occur in academic mesh it becomes complex manually rearranging of schedule, making this process doesn't be the most optimal and efficient, requiring more resources, further delaying the design of schedules.

It is proposed the development of a system divided into modules that optimize the schedule generation process. Two of these modules are specified in this thesis project: Academic Management of Careers, and Availability of Teachers, created and developed under the parameters set by all the working groups in technology and development tools.

In this work are defined the designe and the module characteristics: Its interaction with users, the work they do within UPS Schedule system, the integration performed for correlation with other modules in order to contribute and complement the working of the complete system.

INTRODUCCIÓN

○ Planteamiento del problema

La Universidad Politécnica Salesiana tiene tres sedes a nivel nacional, la sede que se encuentra en la ciudad de Quito está compuesta por tres campus que ofrecen una oferta académica de educación superior humanística y politécnica.

Al iniciar un periodo académico la elaboración de horarios, la administración de información académica de carreras, mallas, materias, materias paracadémicas, niveles y grupos de horarios de estudio (matutino, nocturno) es una tarea que consume recursos, sobretodo de tiempo de quienes se encargan de administrar toda esta información generalmente son los directores de cada carrera.

Del mismo modo los docentes al ingresar su horario disponible para dictar clases toma tiempo, ya que los docentes cumplen diferentes actividades, conjuntamente con las clases, realizan tutorías de trabajos de titulación, investigación y horas destinadas a la administración de las carreras.

Con el desarrollo de los módulos de Gestión académica de carreras y disponibilidad docente, el sistema UPS Schedule podrá administrar la información de mallas, carreras, materias, materias paracadémicas, niveles y grupos de horarios de estudio (matutino, nocturno), también el sistema necesita de un módulo que permita el ingreso de la información acerca de la disponibilidad horaria del docente. Tal como se ha visto se necesita de una herramienta adecuada, que facilite la administración de la información de cada carrera y las horas disponibles del docente, para mejorar los tiempos y la exactitud de los horarios entregados al iniciar un periodo académico.

Para el presente trabajo de titulación se plantea los siguientes objetivos:

- **Objetivos general**

Desarrollar los módulos de gestión académica de carreras y disponibilidad de docentes para el sistema UPS Schedule para la Universidad Politécnica Salesiana sede Quito.

- **Objetivos específicos**
  - Analizar el proceso de disponibilidad horaria del docente en la creación de horarios para el período académico
  - Analizar el proceso general de la gestión académica de mallas, niveles, materias y grupos
  - Diseñar los módulos de: gestión académica de carreras y disponibilidad de horario del docente en su totalidad
  - Programar los módulos utilizando herramientas de software libre
  - Probar el correcto funcionamiento de los módulos
  - Integrar los módulos desarrollados, al sistema UPS Schedule
  - Generar los reportes para los módulos desarrollados

- **Justificación del problema**

En la actualidad, la Universidad Politécnica Salesiana Sede Quito, requiere de una herramienta que designe los horarios, de acuerdo a la disponibilidad horaria de cada docente, este trabajo complementará el proyecto de desarrollo UPS Schedule, que se viene desplegando por módulos a cargo de varios tesistas. La integración de estos módulos, dará como resultado una herramienta práctica y versátil que evitará generar o calcular manualmente los horarios para cada periodo académico.

La correcta funcionalidad del sistema UPS Schedule requerirá el ingreso de la disponibilidad horaria de los docentes, además de la gestión de carreras, lo que involucra la administración de: mallas académicas, niveles, materias y grupos; por tanto se propone el desarrollo del módulo Gestión académica de carreras y el módulo de

Disponibilidad de docentes como solución para la administración de la información académica y disponibilidad horaria de docentes.

- **Alcance del proyecto**

El proyecto define dos módulos que permiten administrar la información que genera cada una las carreras y docentes de la Universidad Politécnica Salesiana sede Quito, los módulos de Gestión académica de carreras y disponibilidad horaria de docentes se integran al sistema UPS Schedule para su funcionamiento completo, para esto se hará uso de herramientas de software libre como: Java, librería Primefaces, PostgreSQL, SQL Power Architect, NetBeans y Jasper Report.

Se establecerá un estándar de nomenclatura y etiquetado para el uso de clases, métodos, variables, tablas y campos que será adoptado por todos los desarrolladores en cada uno de los módulos del sistema UPS Schedule de la Universidad Politécnica Salesiana Sede Quito.

- **Módulo de Gestión académica de carreras**

El módulo Gestión académica de carreras, será parte de la administración central del sistema UPS Schedule, este módulo se encarga de gestionar (crear, modificar, eliminar) toda la información de mallas, carreras, materias académicas, paracadémicas, y niveles de las carreras de la universidad, permite también obtener un reporte en el formato que el usuario elija (PDF, DOC, CSV).

El módulo estará conformado de los siguientes submódulos:

- **Gestión de mallas:** Este submódulo permitirá al usuario: crear, modificar, eliminar, habilitar o deshabilitar las mallas para cada carrera en el período activo.
- **Gestión de materias:** Este submódulo permitirá al usuario: crear, modificar, eliminar, habilitar o deshabilitar las materias para cada carrera en el período vigente.

- **Gestión de niveles:** Este submódulo permitirá al usuario: crear, modificar, eliminar, habilitar o deshabilitar los niveles para cada carrera en el período activo.

En los submódulos de gestión de mallas, materias y niveles se puede obtener un reporte en formato digital los cuales pueden elegir entre un formato PDF, DOC o CSV, es decir el usuario hará uso de la información que visualiza en pantalla según sea necesario.

- **Módulo de Disponibilidad de docentes**

El módulo de Disponibilidad de docentes permitirá ingresar las horas que el docente puede dictar cátedra en la universidad.

El módulo de Disponibilidad de docentes, permite realizar lo siguiente:

- **Disponibilidad:** Este submódulo permitirá al docente gestionar su disponibilidad de tiempo de lunes a domingo, para el período académico vigente. Le permite al docente visualizar las horas y días de la semana sobre lo cual podrá ingresar la hora y día libre para asistir a clases. Una vez que el docente ingrese la información de su disponibilidad horaria se registra esta información con el período vigente (semestre actual). Adicional el docente puede visualizar las horas y los días en los que trabajó el semestre pasado tener una idea de su anterior horario.

En los siguientes capítulos, se muestra el proceso de desarrollo de los módulos: Gestión Académica de carreras y Disponibilidad de docentes.

Dentro del capítulo 2 se describen: El análisis de viabilidad y de requerimientos y el diseño de los módulos, posterior en el capítulo tres se detalla la codificación, lenguajes de programación, arquitectura de los módulos y pruebas que fueron realizadas y para complementar el desarrollo de los módulos planteados se procede con

La integración de los módulos se especifica en el capítulo 4, el proceso de unificación del sistema, además de cómo fue constituida la base de datos, las adaptaciones

necesarias que se debieron realizar para la unificación de los módulos desarrollados por cada equipo de trabajo.

CAPÍTULO 1

ANÁLISIS Y DISEÑO

1.1 Análisis de viabilidad

Aquí se muestra el análisis de viabilidad, donde se analizan los aspectos técnicos, económicos y operaciones del proyecto.

1.1.1 Viabilidad técnica.

El proyecto utilizará una infraestructura de red con servidores levantados en equipos computacionales de los tesistas, al no ser un proyecto de implementación no se incurrirá en gastos de compra de servidores, esto favorece la viabilidad para ejecutar y probar el proyecto desarrollado, a continuación se detalla las características del equipo computacional, las herramientas y lenguajes a ser usados.

Tabla 1. Detalle de los lenguajes o herramientas para el desarrollo del proyecto

LOS MÓDULOS SERÁN DESARROLLADOS CON	
LENGUAJE/HERRAMIENTA	**DESCRIPCIÓN**
Java	Software libre
NetBeans 7.1.2	Herramienta de desarrollo
Glasfish	Servidor de aplicaciones
JasperReport	Herramienta para generar reportes
PostgreSQL	Servidor de base de datos

Elaborado por: Yadira Santillán & Raúl Torres

Tabla 2. Características del computador a usarse

CARACTERISTICAS DEL COMPUTADOR	
ELEMENTO	**CARACTERÍSTICA**
Procesador	Intel Core i5 2.4 GHz
Memoria RAM	4 GB
Sistema operativo	Windows 7

Elaborado por: Yadira Santillán & Raúl Torres

1.1.2 Viabilidad económica.

Los gastos necesarios para el desarrollo de los módulos: Gestión académica de carreras y Disponibilidad de docentes serán asumidos por los tesistas.

En la siguiente tabla se describe con detalle los gastos generados por el proyecto:

Tabla 3. Detalles de la viabilidad económica

MÓDULOS: GESTIÓN ACADÉMICA DE CARRERAS Y DISPONIBILIDAD DE DOCENTES				
ÍTEM	**COSTE C/U**	**CANT.**	**MESES**	**TOTAL**
Servicios básicos	$ 20,00	1	9	$ 180,00
Internet	$20,16	1	9	$ 181,44
Transporte	$2,00	2	9	$ 36,00
Telefonía móvil	$1,00	2	9	$ 18,00
Útiles de oficina	$5,00	2	9	$ 90,00
Ordenador	$800,00	2	1	$ 1600,00
Impresora	$65,00	1	1	$ 65,00
Hora hombre	$5,00	80	9	$ 3600, 00
Derechos de grado	$485,00	2	1	$ 970,00
TOTAL				**$ 6740,44**

Elaborado por: Yadira Santillán & Raúl Torres

1.1.2.1 Análisis de factibilidad TIR/VAN.

Valor actual neto (VAN): Es el valor neto de la inversión, el VAN está calculada en base a una tasa de descuento (TD).

Tasa interna de retorno (TIR): Es la tasa de descuento con que el VAN se vuelve cero, es el indicador que mide la rentabilidad de un proyecto.

Tabla 4. Desarrollo TIR/VAN

FLUJO DE CAJA						
DETALLE						
A. FLUJO DE BENEFICIOS	**AÑO 0**	**AÑO 1**	**AÑO 2**	**AÑO 3**	**AÑO 4**	**AÑO 5**
Valor que no pagará la universidad después de la implementación del sistema	-	5952,00	6201,39	6461,23	6731,95	7014,02
Costo por implementación	-	0,0	0,0	0,0	0,0	0,0
TOTAL FLUJO DE BENEFICIOS	-	5952,00	6201,39	6461,23	6731,95	7014,02
B. FLUJO DE COSTOS						
Gastos de insumos	2170,44					
Gastos administrativos	970,00	60,00	62,51	65,13	67,86	70,71
Gastos tiempo de desarrollo de software (hora)	3600,00	400,00	416,76	434,22	452,42	471,37
Gastos mantenimiento proyecto		320,00	333,41	347,38	361,93	377,10
TOTAL FLUJO DE COSTOS	6740,44	780,00	812,68	846,73	882,21	919,18
A-B	-6740,44	5172,00	5388,71	5614,49	5849,74	6094,85

Elaborado por: Yadira Santillán & Raúl Torres

VAN	$ 13.345,33
TIR	75%

Tabla 5. Análisis del periodo de repago

Tiempo - Periodo de repago				
	M0	Año 1	Año 2	Año 3
	-6740,44	5172,00	5388,71	5614,19
Queda	-----	-1568,44	3820,27	9434,46
La inversión se recupera a los 2 años y 6 meses				

Elaborado por: Yadira Santillán & Raúl Torres

En la tabla 5 se identifica el periodo de repago en el cual se evidencia el tiempo en años y meses en los cuales se recupera la inversión y se empieza a generar un ingreso, en este caso el proyecto no genera ingresos, más bien evita desembolsar dinero de la UPS, se ve de forma clara que a partir de sexto mes del tercer año se recupera la inversión inicial y se empieza a tener un ahorro para la UPS.

Tabla 6. Análisis costo / beneficio

COSTO - BENEFICIO		C/B
No Egresos	16174,90	2,03
Tasa de Rentabilidad	12,33%	
Inversión	6740,44	
Tasa de Interés	20,00%	

Elaborado por: Yadira Santillán & Raúl Torres

La tabla 6 del análisis costo/beneficio muestra el beneficio que se obtiene al hacer una inversión, en este proyecto por cada $1,00 que se invierte se recibe $1,05 de ganancia, lo cual para la UPS es evitar ese desembolso de dicho valor y generar un ahorro.

En base al análisis desarrollado, se demuestra que el proyecto es factible de acuerdo al VAN de $13.345,33 mayor a cero, un indicador TIR mayor a la tasa de descuento 12.33%, que en este caso el TIR es de 75% y un costo/beneficio de $1,05 por cada $1,00 invertido inicialmente lo que vuelve a este proyecto rentable.

1.1.3 Viabilidad operacional.

Las interfaces del proyecto serán diseñadas de manera amigable, interactiva e intuitiva, facilitando el acceso y la interacción del usuario con el sistema. El proyecto optimiza los procesos actuales con los que se administra los datos de carreras, materias, mallas y niveles. Estos procesos son desarrollados de forma manual, dentro de la generación de

horarios y en cada carrera, siendo un inconveniente el relacionar espacios físicos, tiempos, disponibilidad del docente, recursos que son compartidos por las distintas carreras entre los tres campus.

El proyecto busca contribuir con un acceso ágil y rápido a la información de carreras, materias, mallas, niveles y disponibilidad horaria lo cual simplifica procesos y optimiza tiempos de consulta de información relacionada a carreras y horarios de docentes.

El análisis de viabilidad en los aspectos técnico, económico y operacional permite concluir que el desarrollo e implementación de los módulos que conforman este proyecto, lo vuelven viable.

1.1.4 Resumen.

Al concluir el análisis, técnico, económico y operacional de las distintas fases del proyecto se concluye que el proyecto es factible.

El uso de las tecnologías de desarrollo propuestas permite diseñar, construir interfaces amigables, intuitivas para el usuario, así como también el uso de éstas herramientas disminuye el costo de desarrollo e implementación, gracias a que no implican un costo por sus licencias.

Con este sistema se optimiza el ingreso y administración de la información académica y disponibilidad de docentes lo cual apoya directamente al trabajo del director y del docente.

1.2 Análisis de requerimientos

1.2.1 Alcance.

Para el desarrollo de los módulos Gestión académica de carreras y disponibilidad de docentes se requiere de la autentificación de los usuarios para obtener el perfil y así gestionar la navegación entre las interfaces que corresponden a estos módulos.

El módulo Gestión académica de carreras, permite al administrador del sistema organizar la información de las carreras con las acciones de: crear, modificar o eliminar una carrera, materia, y nivel.

El módulo DID, le permite al docente seleccionar, modificar las horas disponibles comprendidas entre los días lunes a sábado en los horarios: de 07:00 a 13:00, como horario matutino, y desde las 13:30 a 21:30, como horario nocturno.

1.2.2 Definiciones, acrónimos y abreviaturas.

- **UPS:** Universidad Politécnica Salesiana
- **HTTP:** Hypertext Transfer Protocol o Protocolo de Transferencia de Hipertexto
- **GAC:** Gestión académica de Carreras
- **DID:** Disponibilidad de Docentes
- **AC:** Administración de Carreras

1.2.3 Visión general.

A continuación se describe las funcionalidades y limitaciones de los módulos de Gestión académica de carreras y disponibilidad horaria del docente.

1.2.4 Descripción general.

1.2.4.1 Perspectivas del producto.

El módulo Gestión académica de carreras y el módulo Disponibilidad de docente se integrarán al sistema UPS Schedule, como una aplicación web que interactuará con los datos que requiere cada carrera.

En el módulo de Disponibilidad del docente, el usuario con perfil docente ingresará las horas disponibles que serán tomadas en cuenta para la elaboración del horario académico.

1.2.4.2 Funciones del producto.

El acceso a los módulos desarrollados están definidos por perfiles que son asignados a los usuarios registrados, estos perfiles restringen el acceso y modificación al módulo de Gestión académica de carreras o al módulo de Disponibilidad del docente.

El módulo de Gestión académica de carreras administra la información de carreras, mallas, materias, niveles, grupos y materias paracadémicas. Cuando el usuario accede a este módulo puede crear, modificar o eliminar la información que administra, adicional puede obtener reportes de las carreras y materias que se encuentran habilitadas para el semestre vigente. En el módulo de Disponibilidad del docente el usuario pude visualizar la disponibilidad horaria del semestre anterior e ingresar la disponibilidad horaria para el nuevo semestre.

1.2.4.3 Características del usuario.

El módulo de Gestión académica de carreas está dirigido al usuario con perfil Administrador quien es el encargado de administrar toda la información que hace referencia a la carrera. El módulo de Disponibilidad del docente está dirigido al usuario con perfil docente en el que puede ingresar las horas disponibles para dictar clases en la semana laboral.

1.2.4.4 Restricciones de diseño.

Las herramientas y plataformas que se utilizan para el desarrollo del proyecto deberán ser software libre.

Los módulos de Gestión académica de carreras y Disponibilidad de docentes dependen del módulo de usuarios y del módulo de permisos los cuales son desarrollados por otros tesistas, el módulo de usuarios permite identificar si el usuario que intenta ingresar al sistema pertenece a la universidad y el tipo de perfil asignado, el módulo de permisos es el encargado de autorizar que las clases que permiten el funcionamiento de los módulos planteados se ejecuten y permitan al usuario administrar la información académica y, en el caso de docentes, ingresar su disponibilidad horaria.

1.2.5 Requerimientos específicos.

1.2.5.1 Interfaces de usuario.

La interfaz será muy amigable e intuitiva, de modo que cualquier usuario pueda utilizarla sin complicación. La interfaz constituye un diseño óptimo orientado y pensado para cubrir las necesidades conceptuales y físicas de los usuarios como por ejemplo:

- Interfaz de usuario 100% web
- Textos claros y específicos que indiquen su funcionamiento interno de forma sencilla y general
- Funciones que faciliten la eliminación, creación y modificación mediante controles que permitan seleccionar el elemento a manipular

1.2.5.2 Interfaces de hardware.

Los módulos deben ser compatibles con arquitecturas X86_64, para la visualización de los usuarios se debe considerar los estándares de resolución 1024 x 768 píxeles.

1.2.5.3 Interfaces de software.

Los módulos interactúan directamente con otros módulos del sistema UPS Schedule

1.2.5.4 Interfaces de comunicaciones.

Los módulos deben trabajar bajo el protocolo HTTP, el acceso será desde internet o desde la intranet de la Universidad Politécnica Salesiana

1.2.6 Requerimientos funcionales.

A continuación se describen los requerimientos funcionales mediante la siguiente plantilla organizada por cada perfil de usuario del sistema.

1.2.6.1 Administrador del sistema.

Tabla 7. Requerimiento funcional 1.1: Gestión de carreras

Descripción	Crear, modificar y eliminar carreras de la sede
Precondición	Sedes existentes en el sistema
Entrada	Datos de la carrera (Nombre, ubicación)
Proceso	Datos modificados en el sistema
Salida	Mensaje de aceptación / Error

Elaborado por Yadira Santillán & Raúl Torres

Tabla 8. Requerimiento funcional 1.2: Gestión de niveles

Descripción	Crear, modificar y eliminar niveles de una carrera
Precondición	Carrera exista en el sistema
Entrada	Datos de los niveles (número de niveles)
Proceso	Datos modificados en el sistema
Salida	Mensaje de aceptación / Error

Elaborado por Yadira Santillán & Raúl Torres

Tabla 9. Requerimiento funcional 1.3: Gestión de materias

Descripción	Crear, modificar y eliminar materias de las carreras
Precondición	Carrera y niveles existentes en el sistema
Entrada	Datos de la materia
Proceso	Datos modificados en el sistema
Salida	Mensaje de aceptación / Error

Elaborado por Yadira Santillán & Raúl Torres

Tabla 10. Requerimiento funcional 1.4: Gestión de grupos

Descripción	Crear, modificar y eliminar grupos
Precondición	Carrera, niveles y periodo existentes en el sistema
Entrada	Datos del grupo (número, nivel asignado)
Proceso	Datos modificados en el sistema
Salida	Mensaje de aceptación / Error

Elaborado por Yadira Santillán & Raúl Torres

Tabla 11. Requerimiento funcional 1.5: Gestión de mallas

Descripción	Crear, modificar y eliminar en una malla
Precondición	Carrera existentes en el sistema
Entrada	Malla vigente
Proceso	Datos modificados en el sistema
Salida	Mensaje de aceptación / Error

Elaborado por Yadira Santillán & Raúl Torres

Tabla 12. Requerimiento funcional 1.6: Reporte de carreras

Descripción	Generar un reporte de carreras
Precondición	Carreras existentes en el sistema
Entrada	Nombres de carrera, nivel, materia y malla
Proceso	El sistema genera un reporte de las carreras
Salida	Reporte de Carrera para visualizar o imprimir

Elaborado por Yadira Santillán & Raúl Torres

Tabla 13. Requerimiento funcional 1.7: Reporte de materias

Descripción	Generar un reporte de materias
Precondición	Materias existentes en el sistema
Entrada	Nombres de carrera, nivel, materia y malla
Proceso	Genera un reporte materias para la sede y carrera seleccionada
Salida	Reporte de Carrera para visualizar o imprimir

Elaborado por Yadira Santillán & Raúl Torres

1.2.6.2 Docente.

Tabla 14. Requerimiento funcional 2.1: Disponibilidad del docente

Descripción	Seleccionar horas disponibles del docente
Precondición	Período académico existente en la carrera
Entrada	Horas a la semana que el docente está disponible para dar clases
Proceso	Datos almacenados en el sistema
Salida	Mensaje de aceptación / Error

Elaborado por Yadira Santillán & Raúl Torres

Tabla 15. Requerimiento funcional 2.2: Modificación en la DID

Descripción	Modificar disponibilidad docente
Precondición	Período académico existente en la carrera, disponibilidad previamente ingresada
Entrada	Horas a la semana que el docente está disponible para dar clases
Proceso	Datos modificados en el sistema
Salida	Mensaje de aceptación / Error

Elaborado por Yadira Santillán & Raúl Torres

Tabla 16. Requerimiento funcional 2.3: Reporte de disponibilidad del docente

Descripción	Generar un reporte de su horario de disponibilidad
Precondición	disponibilidad ingresada
Entrada	Período académico
Proceso	El sistema generará un reporte del horario de disponibilidad del docente
Salida	Reporte de disponibilidad individual para visualizar o imprimir

Elaborado por Yadira Santillán & Raúl Torres

1.2.7 Requisitos de rendimiento.

- Los módulos estarán disponibles e interactuando con los demás módulos del sistema UPS Schedule las 24 horas del día, los 7 días de la semana
- El módulo emitirá notificaciones de información cuando se realice modificación sobre cualquier registro de carreras, mallas, materias, materia paracadémicas, niveles y grupos de estudio (matutino, nocturno)
- Los módulos generan reportes para impresión física y visual
- Deberá soportar al menos 20 usuarios concurrentes

1.3 Diseño del sistema

Se propone la implementación de un sistema informático que esté conformado por módulos. El módulo de Gestión académica de carreras se encargará de administrar la información de todos los datos de las diversas carreras y el módulo de Disponibilidad de docentes permite al docente ingresar las horas disponibles para dictar la cátedra.

Para diseñar el sistema se hace uso del lenguaje de modelamiento unificado (UML) el que permite definir, observar y documentar mediante gráficos todas las partes que conforman los módulos del sistema.

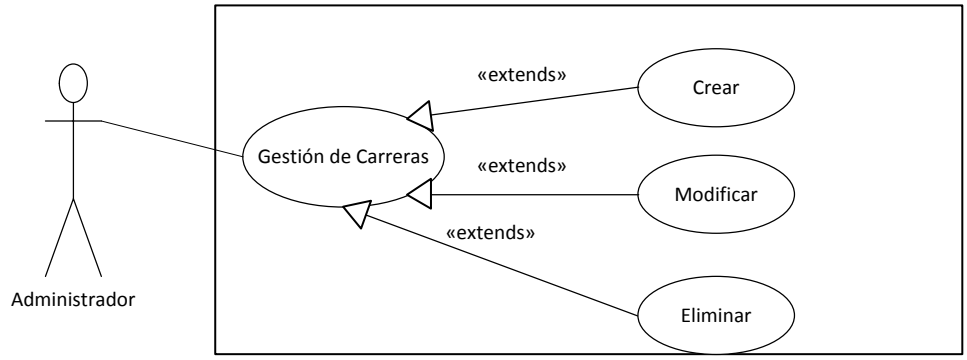
1.3.1 Diagramas de casos de uso.

Con los diagramas de casos de uso se puede identificar claramente la forma y el orden en que el usuario interactúa con el sistema desarrollado.

1.3.1.1 Gestión de carreras.

El usuario administrador puede crear, eliminar o modificar la información de las carreras existentes según el campus seleccionado

Figura 1. Diagrama caso de uso – Gestión de carreras



Elaborado por: Yadira Santillán & Raúl Torres

Tabla 17. Caso de uso: Gestión de carreras

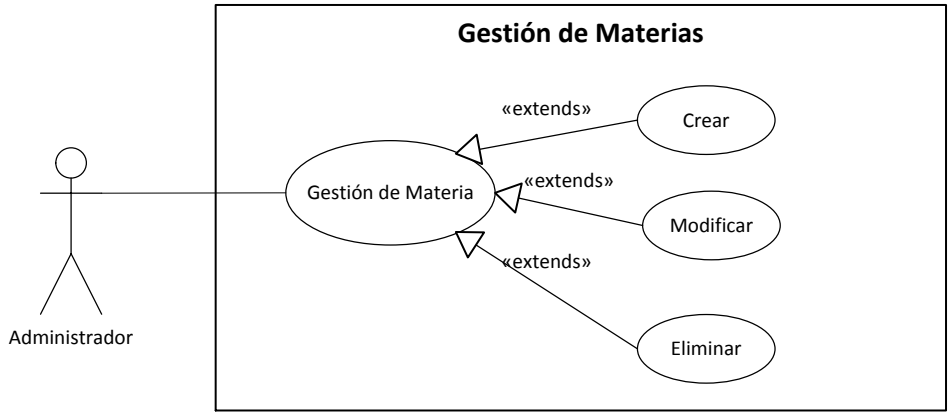
CASO DE USO	GESTIÓN DE CARRERAS
ACTORES	Administrador
PROPÓSITO	Se encarga de crear, modificar y eliminar la información de las carreras de la Universidad Politécnica Salesiana sede Quito
RESUMEN	El usuario puede realizar cambios sobre los datos de las carreras
ACCIÓN DEL ACTOR	**RESPUESTA DEL SISTEMA**
Selecciona el campus para desplegar las carreras existentes	Una vez elegida la carrera según el campus se puede crear, modificar y eliminar las carreras actuales

Elaborado por: Yadira Santillán & Raúl Torres

1.3.1.2 Gestión de materias.

El usuario administrador puede crear, eliminar, modificar la información de las materias existentes según el campus y carrera seleccionado.

Figura 2. Diagrama caso de uso – Gestión de materias



Elaborado por: Yadira Santillán & Raúl Torres

Tabla 18. Caso de uso: Gestión de materias

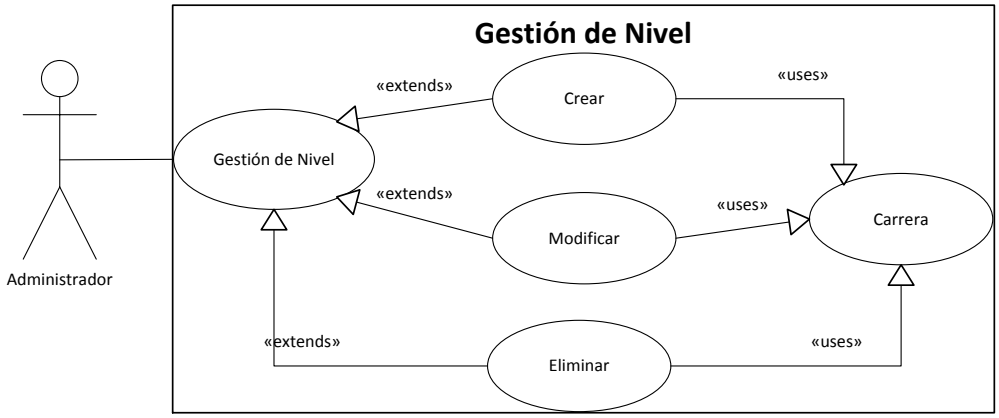
CASO DE USO	GESTIÓN DE MATERIAS
ACTORES	Administrador
PROPÓSITO	Se encarga de crear, modificar y eliminar la información de las materias de la UPS
RESUMEN	El usuario puede realizar cambios sobre los datos de las materias
ACCIÓN DEL ACTOR	**RESPUESTA DEL SISTEMA**
Selecciona el campus, luego selecciona la carrera para desplegar las materias existentes	Una vez elegida el campus y la carrera se muestra las materias filtradas por campus y carreras sobre las cuales se puede crear, modificar y eliminar las materias existentes

Elaborado por: Yadira Santillán & Raúl Torres

1.3.1.3 Gestión de niveles.

El usuario con perfil administrador puede crear, eliminar o modificar la información de los niveles.

Figura 3. Diagrama caso de uso – Gestión de materias



Elaborado por: Yadira Santillán & Raúl Torres

Tabla 19. Caso de uso: Gestión de niveles

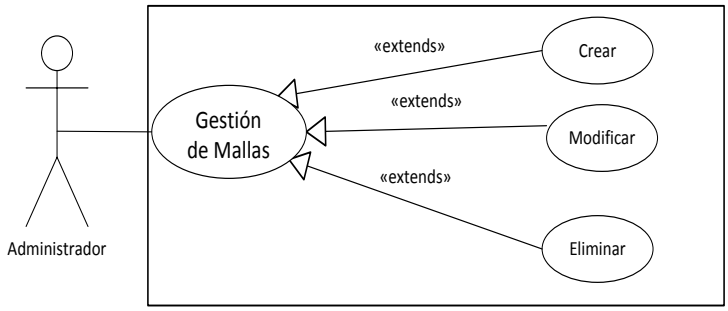
CASO DE USO	GESTIÓN DE NIVELES
ACTORES	Administrador
PROPÓSITO	Se encarga de crear, modificar y eliminar la información de los niveles de la Universidad Politécnica Salesiana sede Quito
RESUMEN	El usuario puede realizar cambios sobre los datos de los niveles
ACCIÓN DEL ACTOR	**RESPUESTA DEL SISTEMA**
Selecciona el nivel a modificar, elige la opción de crear, modificar o eliminar	Crea, modifica o elimina el nivel seleccionado por el usuario

Elaborado por: Yadira Santillán & Raúl Torres

1.3.1.4 Gestión de mallas.

El usuario con perfil administrador puede crear, eliminar o modificar la información sobre las mallas.

Figura 4. Diagrama caso de uso – Gestión de mallas



Elaborado por: Yadira Santillán & Raúl Torres

Tabla 20. Caso de uso: Gestión de mallas

CASO DE USO	GESTIÓN DE MALLAS
ACTORES	Administrador
PROPÓSITO	Se encarga de crear, modificar y eliminar la información de los niveles de la Universidad Politécnica Salesiana sede Quito
RESUMEN	El usuario puede realizar cambios sobre los datos de las mallas
ACCIÓN DEL ACTOR	**RESPUESTA DEL SISTEMA**
Selecciona la malla a modificar, elige la opción de crear, modificar o eliminar	Crea, modifica o elimina la malla seleccionada por el usuario

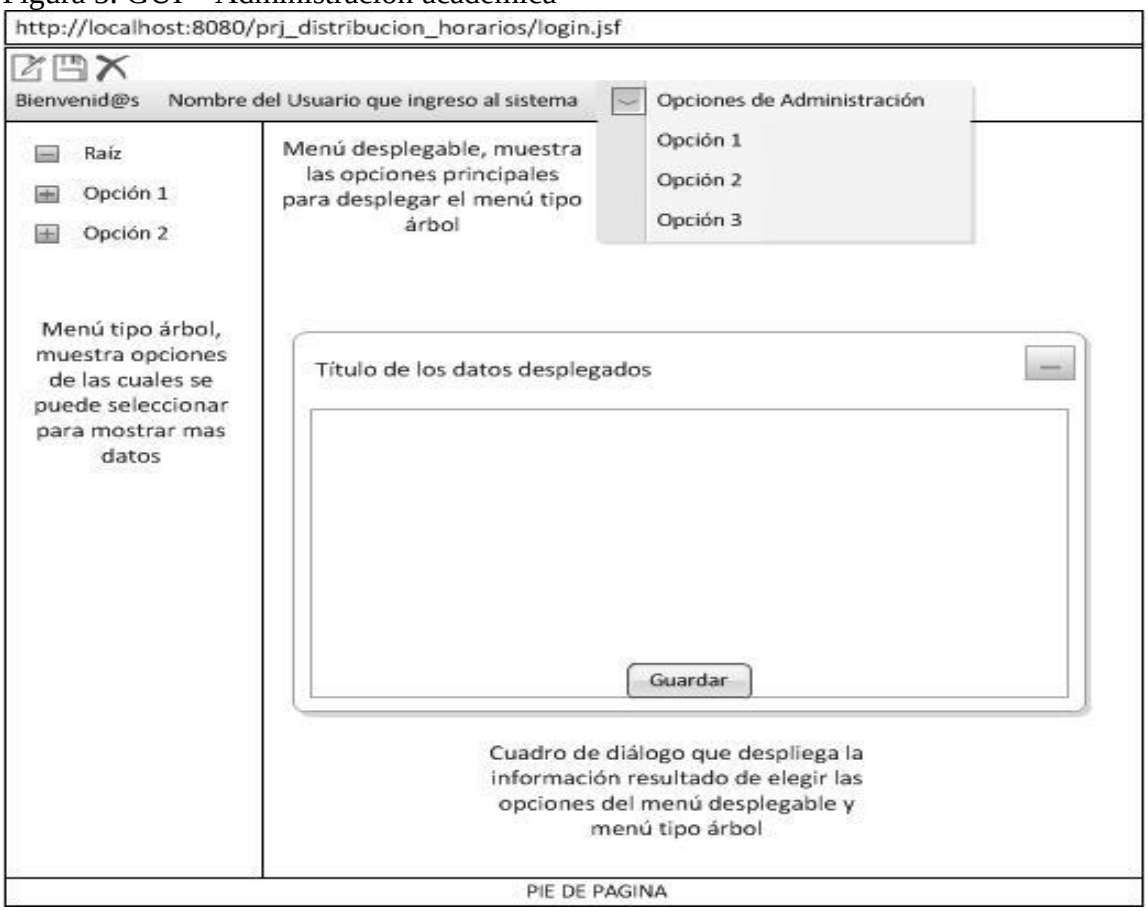
Elaborado por: Yadira Santillán & Raúl Torres

1.3.2 Interfaces de usuario.

Para el diseño de las interfaces de usuario se ha establecido parámetros generales que permitan utilizar la aplicación de manera rápida, simple e intuitiva. El usuario puede acceder a las opciones de los módulos, mediante menús desplegables o por árbol de navegación, facilitando la interacción con el usuario.

Existe también los íconos de acción que permiten crear, eliminar o guardar (después de una modificación), estos íconos se encuentran en la parte superior de la pantalla sobre el mensaje de bienvenida.

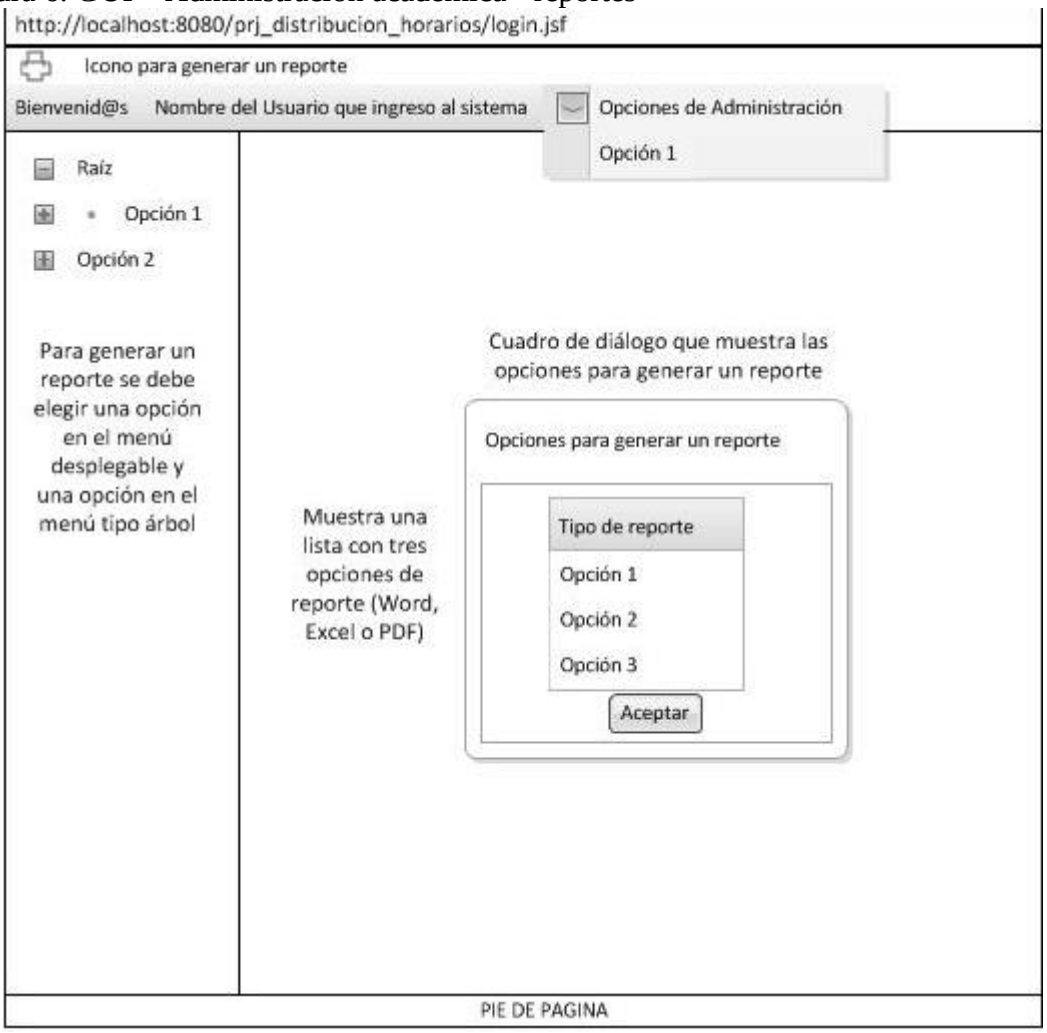
Figura 5. GUI – Administración académica



Elaborado por: Yadira Santillán & Raúl Torres

Cuando el usuario desea generar un reporte tiene un ícono en la parte superior de la pantalla sobre el mensaje de bienvenida, cuando se da clic sobre este ícono, muestra una ventana emergente en la cual se selecciona el reporte a generar.

Figura 6. GUI – Administración académica - reportes

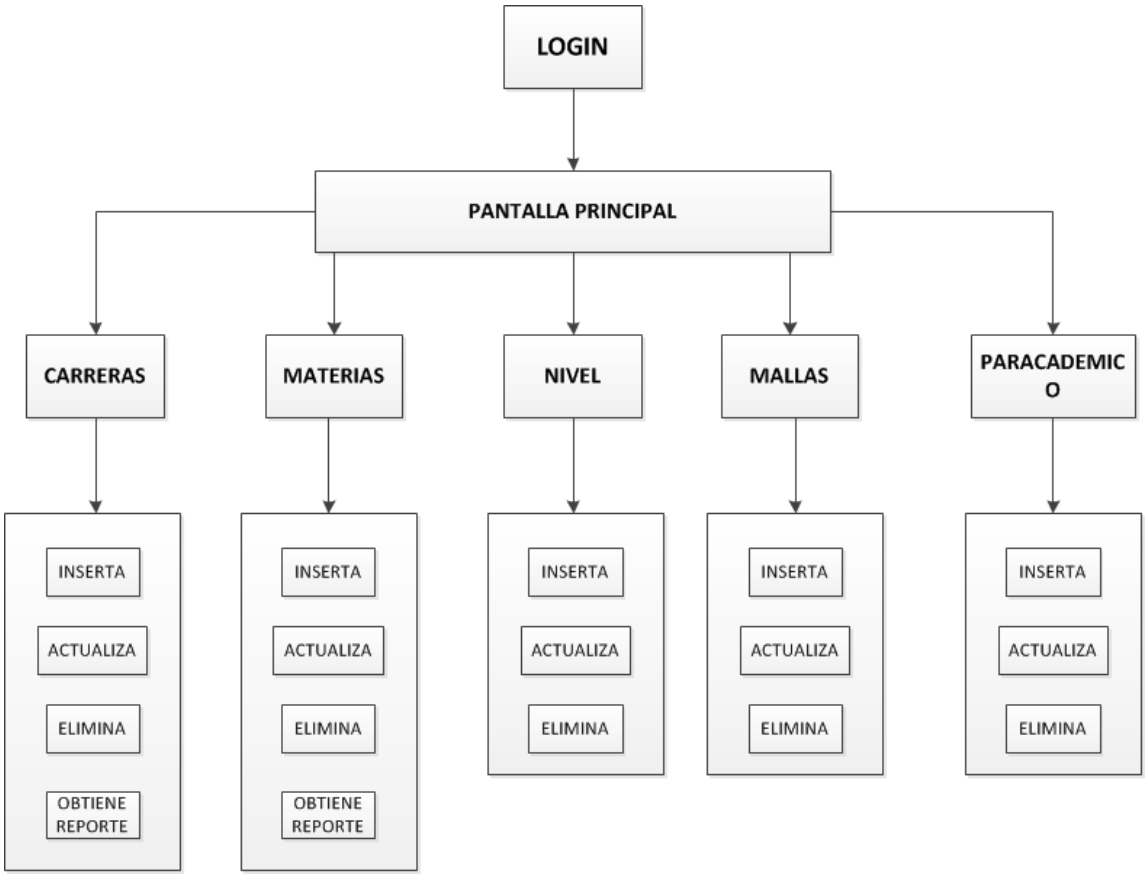


Elaborado por: Yadira Santillán & Raúl Torres

1.3.3 Mapa de navegación.

Se contempla tres perfiles para los usuarios los cuales tienen acceso a la administración académica y otro para el ingreso de la disponibilidad horaria, los usuarios que correspondan al perfil administrador ingresa a la página principal de administración la cual consta de un menú que contiene opciones para administrar la información de carreras, materias, niveles, mallas y materias paracadémicas, adicional se puede generar un reporte de la opción elegida, a continuación se describe gráficamente mediante un mapa de navegación las funciones del usuario con perfil administrador.

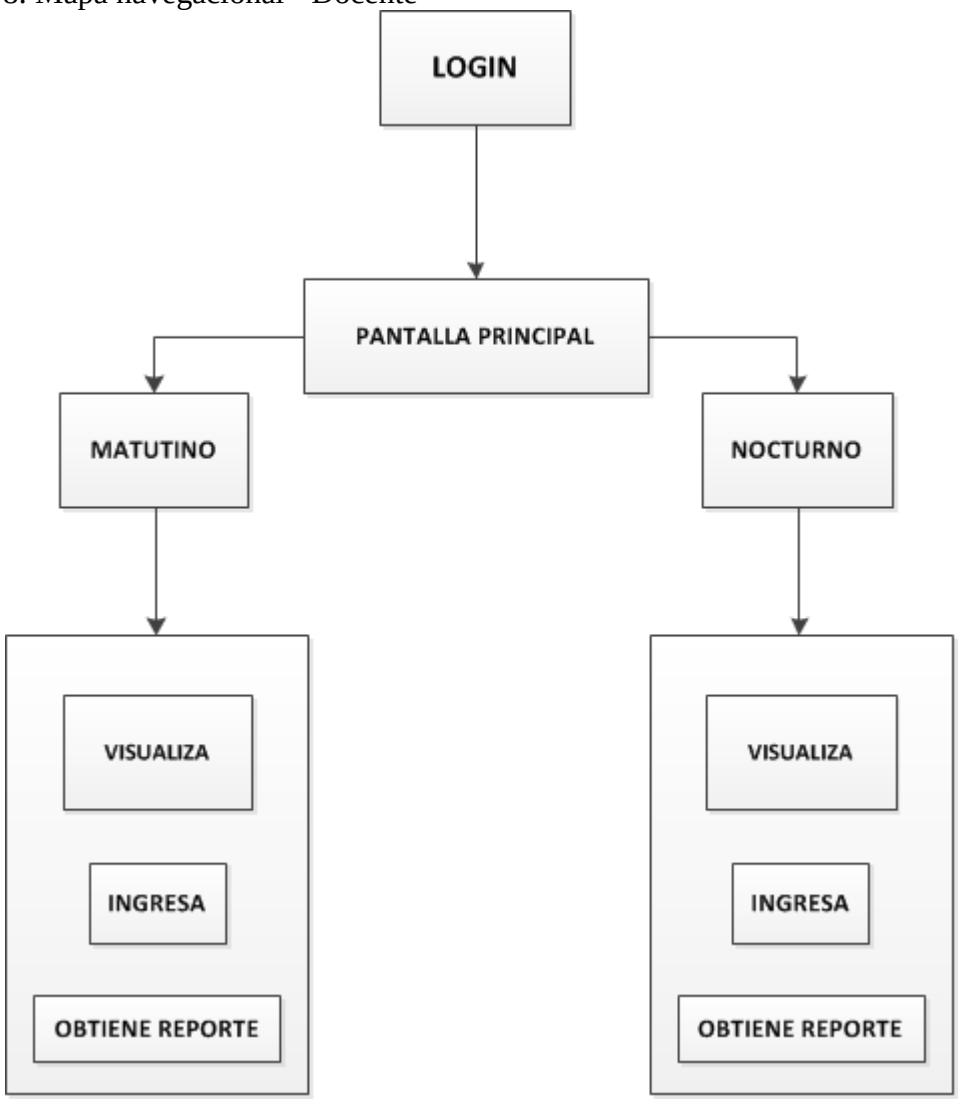
Figura 7. Mapa navegacional - Administrador



Elaborado por: Yadira Santillán & Raúl Torres

El usuario con perfil docente puede visualizar el horario del semestre anterior y una plantilla para que pueda ingresar el nuevo horario para el semestre actual, a continuación se describe mediante un mapa de navegación el usuario con perfil docente:

Figura 8. Mapa navegacional - Docente

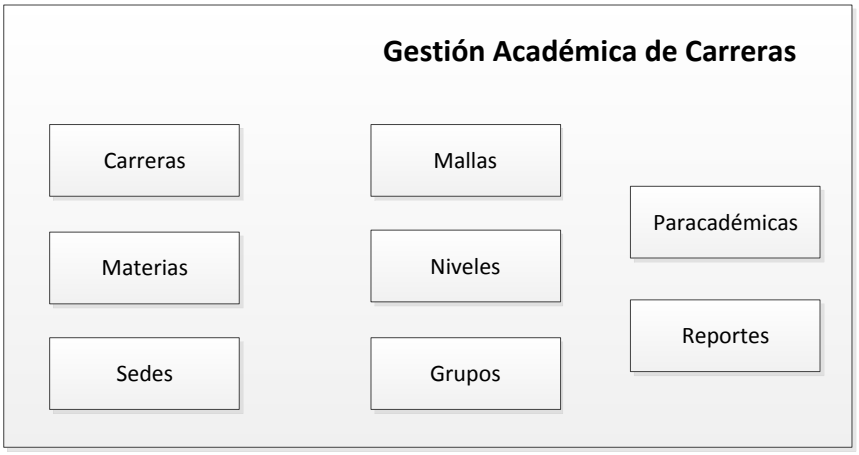


Elaborado por: Yadira Santillán & Raúl Torres

1.3.4 Diagrama de bloques general.

En base al análisis de los requerimientos levantados se define que el sistema se conforma de los siguientes módulos: Gestión académica de carreras y disponibilidad de docentes. Cada uno de estos módulos interactúa con pequeños submódulos que cumplen con los requerimientos del usuario.

Figura 9. Diagrama de bloques – Gestión académica de carreras



Elaborado por: Yadira Santillán & Raúl Torres

Figura 10. Diagrama de bloques – Disponibilidad de docentes

Elaborado por: Yadira Santillán & Raúl Torres

1.3.5 Base de datos.

La base de datos se diseñó en base a los requerimientos levantados para el desarrollo del GAC y el módulo DID en el cual consta las tablas necesarias para almacenar todos los datos relacionados a carreras, docentes y las relaciones entre cada una de las tablas, las que se describen de la siguiente manera:

Tabla 21. Descripción de las tablas utilizadas

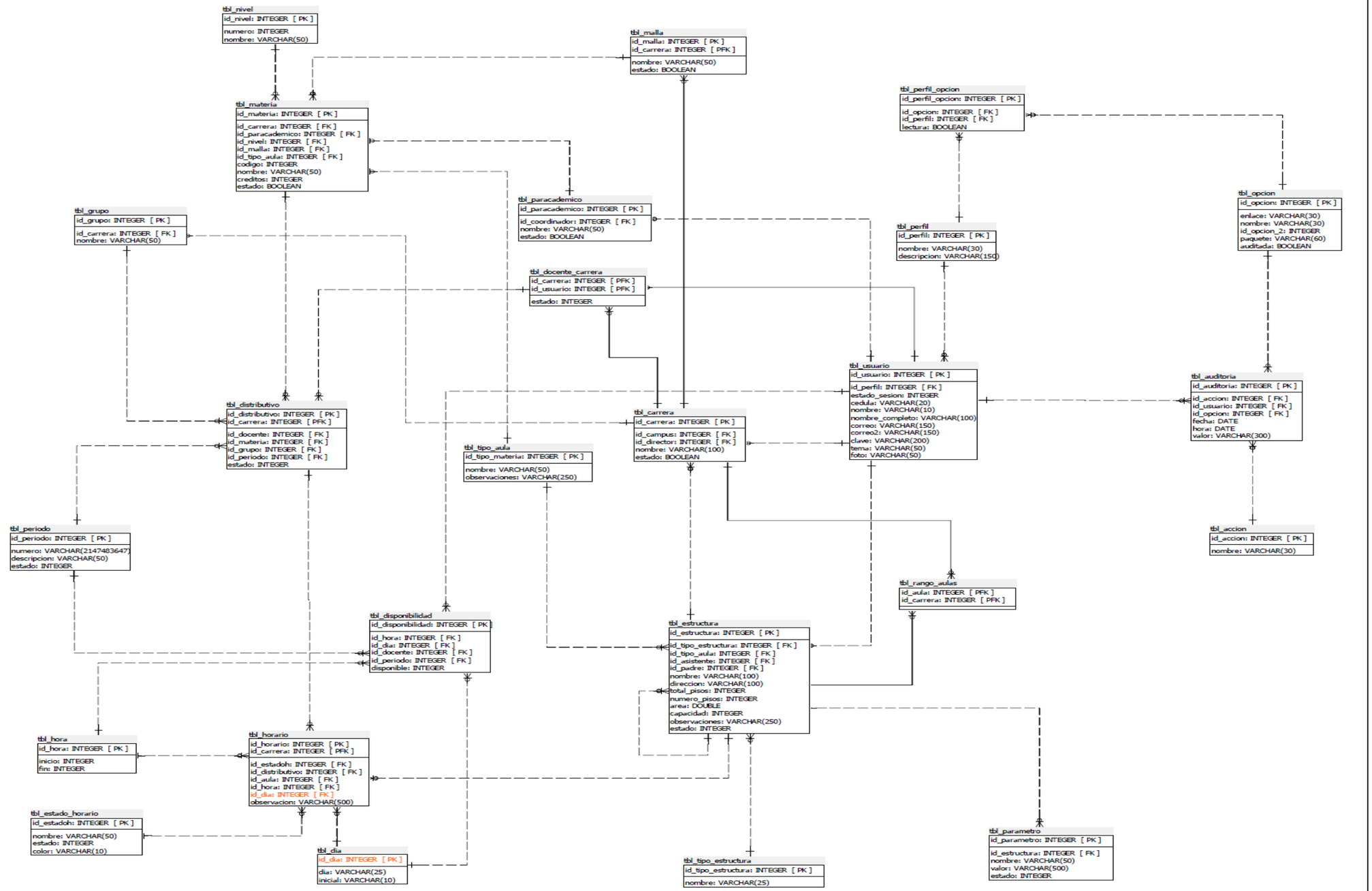
TABLA	DESCRIPCIÓN
tbl_carrera	Almacena la información de las carreras de la sede Quito
tbl_materia	Almacena la información de las materias de la sede Quito
tbl_nivel	Almacena la información de los niveles de la sede Quito
tbl_malla	Almacena malla vigente y mallas antiguas
tbl_paracademico	Almacena información de materias que no pertenecen a la facultad
tbl_tipo_aula	Almacena información del tipo de aula
tbl_estructura	Almacena información de la ciudad, sede y campus
tbl_tipo_estructura	Identificar tipo de estructura: sedes y campus
tbl_usuario	Almacena la información de los usuarios del sistema
tbl_perfil	Almacena la información de los usuarios del sistema
tbl_disponibilidad	Almacena información de las horas disponibles del docente para trabajar
tbl_dia	Almacena información de los días de la semana
tbl_hora	Almacena información de las horas clases de la sede Quito
tbl_periodo	Almacena períodos académicos
tbl_opcion	Define el menú y asocia a la clase Java a utilizarse

Elaborado por: Yadira Santillán & Raúl Torres

1.3.5.1 Diagrama de base de datos.

El diagrama conceptual de la base de datos resume las tablas usadas para el desarrollo de los módulos GAD y DID.

Figura 11. Diagrama Conceptual de la base de datos



Elaborado por: Yadira Santillán & Raúl Torres

1.3.5.2 Diccionario de datos.

Tabla 22. Diccionario de datos – tbl_opcion

Define el menú y asocia a la clase Java a utilizarse según la elección en el menú				
NOMBRE	**VARIABLE**	**NULO**	**¿PK?**	**COMENTARIO**
id_opcion	Integer	Sí	Sí	Código primario de la tabla
enlace	character varying(30)	Sí	No	Nombre de la clase a la cual se enlace la opción
nombre	character varying(30)	Sí	No	Es el nombre visual de la opción en el menú
id_opcion_2	Integer	No	No	Hace referencia al menú principal
paquete	character varying(60)	No	No	Se almacena el nombre del paquete en donde se encuentra la clase ingresada en la opción enlace
auditada	Boolean	Sí	No	Valor booleano para saber si la clase va a ser auditada

Elaborado por: Yadira Santillán & Raúl Torres

Tabla 23. Diccionario de datos – tbl_usuario

Almacena la información de los usuarios del sistema				
NOMBRE	**VARIABLE**	**NULO**	**¿PK?**	**COMENTARIO**
id_usuario	Integer	Sí	Sí	Clave primaria de la tabla tbl_sis_usuario
id_perfil	Integer	Sí	No	Clave primaria de la tabla tbl_sis_perfil
estado_sesion	Integer	Sí	No	Identifica si la sesión del usuario esta activa o no
cedula	character varying(20)	Sí	No	Cédula del usuario
nombre	character varying(10)	Sí	No	Es el nombre con el que se realiza el login (inicio de sesión)
nombre_completo	character varying(100)	Sí	No	Es el nombre real del usuario, servirá para generación de reportes y como información complementaria
correo	character varying(150)	No	No	Es el correo institucional del usuario
correo2	character varying(150)	No	No	Es el correo personal del usuario
clave	character varying(100)	Sí	No	Es la clave con la que se realiza el login (inicio de sesión)
tema	character varying(50)	Sí	No	Tiene la información del tema que personaliza la pantalla del usuario
foto	character varying(50)	No	No	Almacena la foto del usuario

Elaborado por: Yadira Santillán & Raúl Torres

Tabla 24. Diccionario de datos – tbl_perfil

Almacena la información de los usuarios del sistema				
NOMBRE	**VARIABLE**	**NULO**	**¿PK?**	**COMENTARIO**
id_perfil	integer	Sí	Sí	Clave primaria de la tabla tbl_perfil
nombre	character varying(30)	Sí	No	Es el nombre del perfil, por ejemplo: administrador, docente, etc.
descripcion	character varying(150)	No	No	Es la descripción del perfil, por ejemplo: El administrador tiene todos los permisos.

Elaborado por: Yadira Santillán & Raúl Torres

Tabla 25. Diccionario de datos – tbl_nivel

Almacena la información de los niveles de la sede Quito				
NOMBRE	**VARIABLE**	**NULO**	**¿PK?**	**COMENTARIO**
id_nivel	integer	Yes	Yes	Identificador del nivel
numero	integer	Yes	No	Número de nivel

Elaborado por: Yadira Santillán & Raúl Torres

Tabla 26. Diccionario de datos – tbl_carrera

Almacena la información de las carreras de la sede Quito				
NOMBRE	**VARIABLE**	**NULO**	**¿PK?**	**COMENTARIO**
id_carrera	integer	Yes	Yes	Clave primaria, identificador de la carrera
id_campus	integer	Yes	No	Identificador del campus al cual pertenece la carrera
id_director	integer	Yes	No	Clave foránea de la tabla tbl_usuario, representa al director de carrera
nombre	character varying(100)	Yes	No	Nombre de la carrera
color	character varying(10)	Yes	No	Color de la carrera
estado	integer	Yes	No	Estado de la carrera, activa o no activa

Elaborado por: Yadira Santillán & Raúl Torres

Tabla 27. Diccionario de datos – tbl_dia

Almacena información de los días de la semana				
NOMBRE	**VARIABLE**	**NULO**	**¿PK?**	**COMENTARIO**
id_dia	integer	Yes	Yes	Identificador del día
dia	character varying(25)	Yes	No	Nombre del día
inicial	character varying(2)	Yes	No	Inicial del día

Elaborado por: Yadira Santillán & Raúl Torres

Tabla 28. Diccionario de datos – tbl_materia

Almacena la información de las materias de la sede Quito				
NOMBRE	**VARIABLE**	**NULO**	**¿PK?**	**COMENTARIO**
id_materia	integer	Yes	Yes	Clave primaria de la materia
id_tipo_aula	integer	Yes	No	Identificador de aula, relaciona la materia con el aula
id_paracademico	integer	No	No	Si este campo es NULL, entonces la materia es regular o normal
id_nivel	integer	Yes	No	Identifica el nivel al que pertenece la materia
id_malla	integer	Yes	No	Identifica la malla a la que corresponde
codigo	integer	Yes	No	Código interno de la UPS
nombre	character varying(50)	Yes	No	Nombre de la materia
creditos	integer	Yes	No	# de créditos de la materia
estado	integer	Yes	No	Materia, activa o no activa
id_carrera	integer	Yes	No	Identificador de la carrera, relaciona la materia con la carrera

Elaborado por: Yadira Santillán & Raúl Torres

Tabla 29. Diccionario de datos – tbl_hora

Almacena información de las horas clases de la sede Quito				
NOMBRE	**VARIABLE**	**NULO**	**¿PK?**	**COMENTARIO**
id_hora	integer	Yes	Yes	Identificador de la hora
inicio	integer	Yes	No	Hora inicial clase
fin	integer	Yes	No	Hora final clase

Elaborado por: Yadira Santillán & Raúl Torres

Tabla 30. Diccionario de datos – tbl_estructura

Almacena información de las horas que el docente tiene disponible para trabajar				
NOMBRE	**VARIABLE**	**NULO**	**¿PK?**	**COMENTARIO**
id_disponibilidad	integer	Yes	Yes	Identificador de disponibilidad
id_hora	integer	Yes	No	Identificador de la hora
id_dia	integer	Yes	No	Identificador de día
id_docente	integer	Yes	No	Clave primaria de la tabla tbl_usuario. Representa al docente
id_periodo	integer	Yes	No	Identificador del período vigente
disponible	boolean	No	No	Estado (activo o no)

Elaborado por: Yadira Santillán & Raúl Torres

Tabla 31. Diccionario de datos – tbl_disponibilidad

Almacena información de la ciudad, sede y campus				
NOMBRE	**VARIABLE**	**NULO**	**¿PK?**	**COMENTARIO**
id_estructura	integer	Yes	Yes	Identificador de la estructura
id_tipo_estructura	integer	No	No	Clave foránea perteneciente a la tbl_tipo:estructura
id_tipo_aula	integer	No	No	Clave foránea de la tabla tbl_tipo_aula
id_asistente	integer	No	No	
id_padre	integer	No	No	Clave foránea para relación recursiva
nombre	character varying(100)	Yes	No	Nombre a mostrarse
direccion	character varying(100)	No	No	Ubicación física del lugar
total_pisos	integer	Yes	No	Total de pisos
numero_pisos	integer	Yes	No	Número de pisos
area	double precision	Yes	No	Dimensión
capacidad	integer	Yes	No	Capacidad
observaciones	character varying(250)	Yes	No	Comentarios
estado	integer	Yes	No	Activa o no activa

Elaborado por: Yadira Santillán & Raúl Torres

CAPÍTULO 2

CODIFICACIÓN Y PRUEBAS

2.1 Lenguajes de programación, almacenamiento y herramientas

Las herramientas para el desarrollo de los módulos planteados y para el almacenamiento de información son libres sin costos de licenciamiento, a continuación se hace una breve descripción de cada una de ellas.

2.1.1 Java.

Lenguaje de programación desarrollado por Sun Microsystems, una de sus características principales es su portabilidad, esto se consigue mediante un entorno de ejecución para los programas compilados. Este entorno se denomina Java Runtime Enviroment (JRE), este entorno es gratuito y está disponible para los principales sistemas operativos. Esto asegura que el mismo programa Java pueda ejecutarse en Windows, Mac OS, Linux o Solaris sin realizar ningún tipo de modificación (Guevara, 2009).

2.1.1.1 Primefaces.

Primefaces es un conjunto de componentes Java Server Faces (JSF) de fuente abierta con varias extensiones como:

- Conjunto enriquecido de componentes (editor HTML, de diálogo, de autocompletar, gráficos, entre otros).
- Posee un kit móvil de interfaz de usuario para crear aplicaciones Web móviles para dispositivos de mano.
- Acceso a documentación (Çivici, 2012).

PrimeFaces tiene soporte Ajax el cual es transparente para el desarrollador, aunque para activarlo deben utilizarse atributos específicos en cada uno de los componentes para lanzar un método del servidor y para indicar los componentes a

actualizar. Otra de las ventajas en comparación con otras librerías es que PrimeFaces cuenta con más de 100 componentes OpenSource, algunos de muy alta complejidad. Adicional PrimeFaces cuenta con un kit para crear interfaces Web para teléfonos móviles (Cruz, 2012).

2.1.1.2 Java Server Faces (JSF).

La tecnología Java Server Faces maneja interfaces de usuario las cuales residen del lado del servidor (Torrijos).

Java Server Faces (JSF) permite desarrollar rápidamente aplicaciones de negocio dinámicas en las que toda la lógica de negocio se implementa en Java, o es llamada desde Java, creando páginas para las vistas muy sencillas. La principal función del controlador JSF es asociar a las pantallas, clases Java que recogen la información introducida y que disponen de métodos que respondan a las acciones del usuario. JSF resuelve de manera muy sencilla y automática muchas tareas como:

- Mostrar datos al usuario en cajas de texto y tablas.
- Recoger los datos introducidos por el usuario en los campos del formulario
- Controlar el estado de los controles del formulario según el estado de la aplicación, activando, ocultando o añadiendo y eliminando controles y demás elementos.
- Realizando validaciones y conversiones de los datos introducidos por el usuario.
- Rellenando campos, listas, combos y otros elementos a medida que el usuario va interactuando con la pantalla.

Controlando los eventos que ocurren en los controles (pulsaciones de teclas, botones y movimientos del ratón) (Almirón, 2009).

2.1.2 NetBeans 7.1.2.

El IDE Netbeans, es un proyecto iniciado por Sun MicroSystems, quienes son adquiridos el 20 de abril de 2009 por Oracle Corporation; Netbeans es un entorno de desarrollo integrado de aplicaciones de código abierto y libre, tanto para sistemas operativos basados en Windows, Linux, Solaris, Mac, este entorno de desarrollo permite programar aplicaciones web, empresariales, de escritorio, móviles, utilizando las plataformas Java, html5, php, c/c++ (Corporation, 2013).

2.1.2.1 Glassfish.

GlassFish es un servidor de aplicaciones desarrollado por Sun Microsystems que implementa las tecnologías definidas en la plataforma Java EE y permite ejecutar aplicaciones que siguen esta especificación. Este tipo de servidores proporciona generalmente gran cantidad de funcionalidades built in de forma transparente al usuario de manera que no sea necesario escribir código fuente. Estas funcionalidades son posibles ya que los componentes se ejecutan dentro del contenedor en un espacio de ejecución virtual llamado dominio de ejecución. Su función principal es la de interponerse entre las llamadas que se hacen a los métodos de los beans y las implementaciones de los mismos, de modo que, entre otras cosas, puede hacer las comprobaciones para verificar si el usuario que llama al método tiene los permisos adecuados, antes de llamarlo.

Una de las características importantes de Glassfish es que dispone de una arquitectura modular por lo que se puede descargar e instalar solamente los módulos que se necesiten para las aplicaciones de modo que minimiza el tiempo de inicio, consumo de memoria y uso de espacio en disco. Basándose en el modelo de componentes dinámico y completo para Java OSGi (Open Services Gateway Initiative), las aplicaciones y/o componentes de Glassfish pueden ser remotamente instalados, iniciados, actualizados, etc. sin necesidad de reiniciar el servidor (Manchado, 2010).

2.1.3 PostgreSQL.

PostgreSQL es un sistema de gestión de bases de datos objeto-relacional, distribuido bajo licencia BSD y con su código fuente disponible libremente. PostgreSQL utiliza un modelo cliente/servidor y usa multiprocesos en vez de multihilos para garantizar la estabilidad del sistema. Un fallo en uno de los procesos no afectará el resto y el sistema continuará funcionando (Rafaelma, 2010).

2.2 Codificación

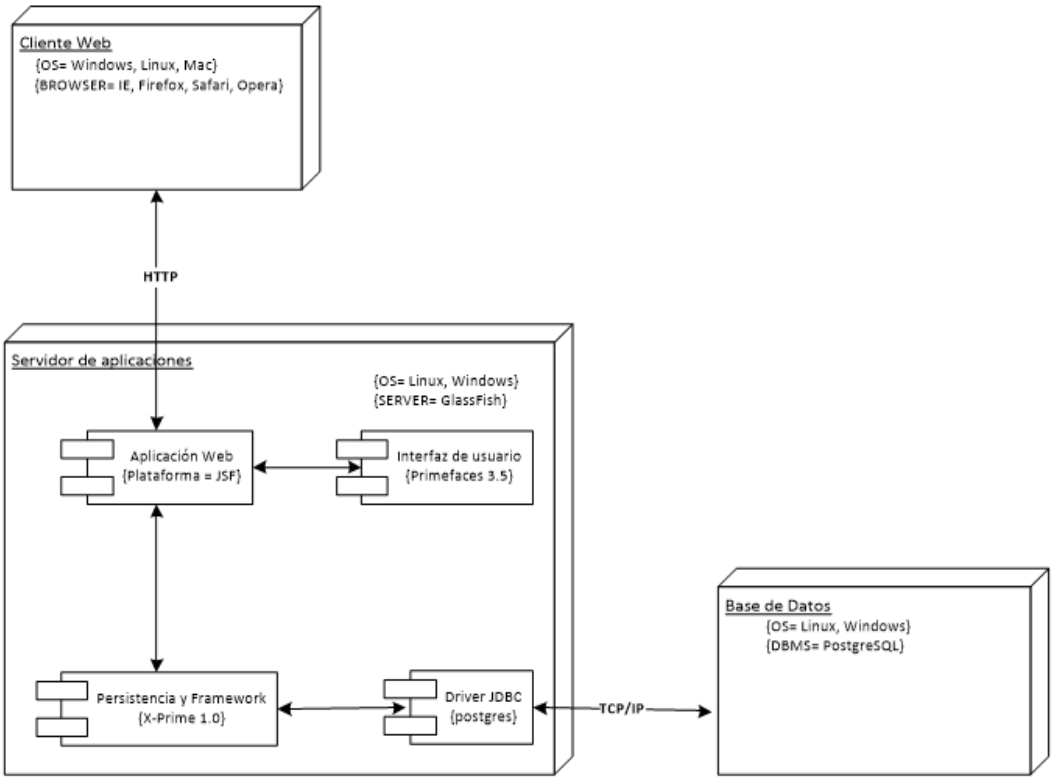
A continuación se describe las partes más importantes del código desarrollado, se explica el framework utilizado, la forma como se realiza la conexión a la base de datos, como se administra la información académica, la disponibilidad horaria y la forma en la que se obtienen los reportes de carreras y materias, también se explica cómo interactúa el gestor de base de datos, el servidor de aplicación y el cliente web.

2.2.1 Arquitectura del sistema.

Para almacenar toda la información se usa como gestor de base de datos PostgreSQL, para el desarrollo de la aplicación web se usa la tecnología JSF y para la interfaz gráfica de usuario se elige la librería de Java Primefaces.

La interacción de los diferentes componentes del sistema se describe a través de un diagrama de despliegue en la figura 12.

Figura 12. Diagrama de despliegue

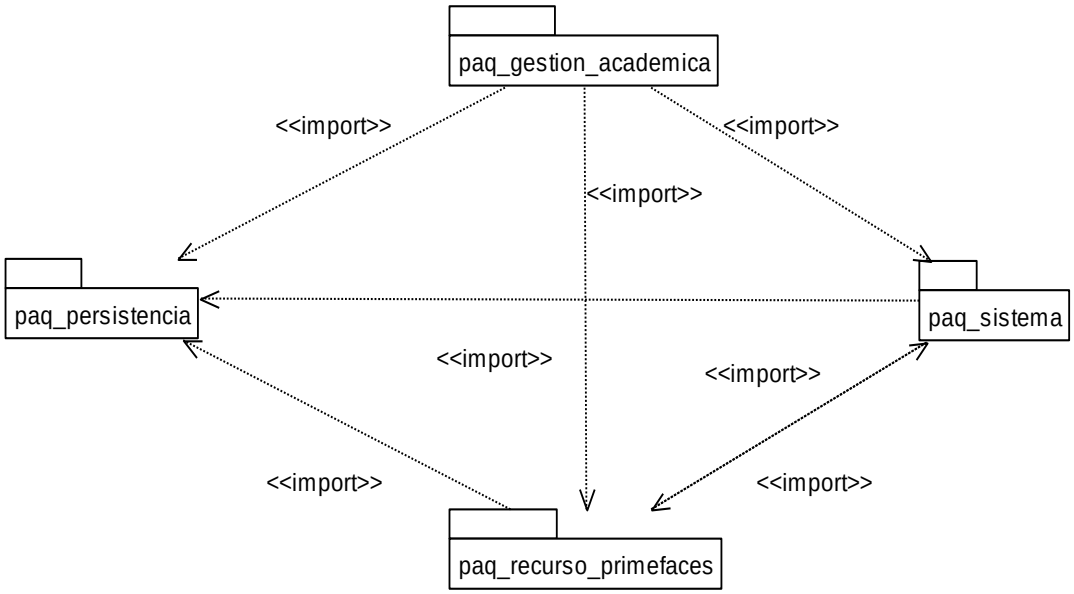


Elaborado por: Danilo Oviedo & José Granda (Tesistas UPS Schedule)

2.2.1.1 Diagrama de paquetes.

Para el desarrollo de este proyecto se ha definido agrupar las clases en diferentes paquetes despendiendo de la función que cumplen, a continuación se describe en el siguiente diagrama:

Figura 13. Diagrama de paquetes



Elaborado por: Yadira Santillán & Raúl Torres

A continuación se muestra parte del código que realiza la persistencia entre la base de datos y la aplicación, adicional se detalla la clase que recupera la información desde la base de datos y muestra en la interfaz gráfica de usuario usando componentes primefaces.

2.2.1.2 Framework X-PRIME V. 1.0.

Se detalla el uso del framework X-PRIME el cual fue desarrollado en lenguaje JSF por Eduardo Pacheco (Tesista UPS Schedule), se basa principalmente en la reutilización de código y software libre, es una implementación y adaptación de PrimeFaces con el objetivo de permitir al desarrollador facilitar su trabajo al momento de crear una aplicación WEB.

X-PRIME permite un desarrollo rápido de los módulos planteados, donde se reutilizó el código ya creado tanto para el acceso a la base de datos como para el diseño de la interfaz gráfica de usuario.

2.2.2 Paquete: paq_persistencia.

2.2.2.1 Clase cla_conexion.

La clase cla_conexion es de gran utilidad para el desarrollo de los módulos, ya que interactúa directamente con la base de datos lo que permite extraer e ingresar información mediante un pool de conexiones JSF. A continuación se muestra parte del código en donde se describe como se obtiene la conexión a la base de datos, mapeando los pool de conexiones dentro del entorno y mediante un ResuslSet realizar las acciones.

```
public cla_conexion() {

        fabrica = Persistence.createEntityManagerFactory (unidad_persistencia);
        manejador = fabrica.createEntityManager ();
        manejador.setFlushMode (FlushModeType.COMMIT);

        try {

        Context lcon_ctx = new InitialContext ();
        DataSource ldso_das = (DataSource) lcon_ctx. Lookup ("pool_postgres");
        conexion_postgres = ldso_das.getConnection ();
        conexion_postgres.setAutoCommit (false);
        conexion_postgres.setTransactionIsolation(Connection.TRANSACTION_READ_COMMI
TTED);
        LOGGER.log (Level.INFO, "Conectado a pool postgres");

        } catch (Exception ex) {
                LOGGER.log (Level.SEVERE, "Ocurrió un error. Causa--> {0}", ex);
                sis_soporte.obtener_instancia_soporte().crear_error(ex.getMessage(), "Clase
            Conexión");
        }
}
```

2.2.3 Paquete: paq_recurso_primefaces.

Este paquete contiene varias clases que permiten estructurar las interfaces gráficas de usuario con varios componentes PrimeFaces, las clases principales que se usa para el desarrollo del proyecto son la clase pf_tabla, pf_arbol, pf_barra_boton, pf_menu, pf_panel y pf_reporte.

Estas clases son reutilizadas en cada uno de los módulos desarrollados, de forma que el código no se vuelve a crear nuevo si no se reutiliza bajo la necesidad del desarrollo.

2.2.3.1 Clase pf_tabla.

Esta clase es muy útil para la presentación de la información al usuario, sirve como capa de persistencia entre la base de datos y la aplicación web, es una implementación del componente DataTable de la librería PrimeFaces por lo que se puede hacer uso de todas sus opciones, una de esas opciones es la edición de registros que son presentados, listar la información de forma ordenada y permite definir el número de registros a ser mostrados.

El constructor de la clase pf_tabla permite definir la tabla sobre la cual se va a realizar la consulta y hace un llamado al método formar_columnas el cual es el que define la forma en el que se muestra la información.

```
public void setTabla(String tabla, String campoPrimaria) {
        this.tabla = tabla;
        this.campoPrimaria = campoPrimaria;
        this.sql = "SELECT * FROM " + tabla;
        formar_columnas ();
}
```

El método formar_columnas () realiza una consulta en la base de datos de la tabla que se va a construir, obtiene el campo y lo agrega a un arreglo de columnas las cuales luego se van a mostrar junto al arreglo de filas en forma de una tabla, esta tabla será editable.

```
private void formar_columnas() {
      try {
        icon_conexion.setSentencia (icon_conexion.getConexion_postgres ().createStatement ());
        ResultSet resultado = icon_conexion.getSentencia ().executeQuery (getSql ());
        ResultSetMetaData rsm_metadata = resultado.getMetaData ();
        columnas = new per_columna [rsm_metadata.getColumnCount()];
        for (int i = 1; i <= rsm_metadata.getColumnCount(); i++) {
          per_columna columna_nueva = new per_columna ();
          if (rsm_metadata.isNullable (i) == 0) {
            columna_nueva.setRequerida (true);
          } else {
            columna_nueva.setRequerida (false);
          }
          columna_nueva.setNombre (rsm_metadata.getColumnName (i).toUpperCase ());
          if (columna_nueva.getNombre().equalsIgnoreCase(campoPrimaria)) {
            columna_nueva.setLectura (true);
          }
          columna_nueva.setNombre_visua l(rsm_metadata.getColumnName(i).toUpperCase());
          columna_nueva.setLongitud (rsm_metadata.getPrecision (i));
          columna_nueva.setAncho (rsm_metadata.getPrecision (i));
          columna_nueva.setDecimales (rsm_metadata.getScale (i));
          columna_nueva.setTipo_Java (rsm_metadata.getColumnClassName (i));
          columna_nueva.setTipo (rsm_metadata.getColumnTypeName (i).toUpperCase ());
           if (columna_nueva.getTipo_Java ().equals("Java.sql.Date")) {
            columna_nueva.setControl ("pf_calendario");
          }
          if (columna_nueva.getTipo_Java().equals("Java.sql.Time")) {
            columna_nueva.setContro l("pf_hora");
          }
          if (columna_nueva.getTipo_Java().equals("Java.lang.Boolean")) {
            columna_nueva.setControl ("pf_revisado");
          }
          columnas [(i - 1)] = columna_nueva;
        }
        resultado.close ();
        icon_conexion.getSentencia ().close ();
      } catch (Exception e) {
          sis_soporte.obtener_instancia_soporte ().crear_error ("No se pudo formar las columnas de la
          tabla " + tabla + ": " + e.getMessage (), "en el método formarColumnas ():");
        System.out.println ("ERROR formarColumnas ():" + e.getMessage ());
      }
    }
```

2.2.3.2 Clase pf_menu.

La clase "pf_menu" presenta un menú estilizado con características propias de un menú PrimeFaces, esta clase hereda los atributos del componente Menú de PrimeFaces. Para mostrar los ítems del menú en el sitio web, el constructor de la pf_menu realiza una consulta de la tabla "tbl_opcion" con la condición del perfil de usuario, es decir se identifica primero el tipo de usuario que está en ese momento haciendo uso del sistema. El resultado de esta consulta se guarda en una variable del tipo lista para luego ser mostrado en la aplicación web, adicional se da la característica gráfica a ser mostrada al usuario.

```
public pf_menu() {
        pf_item_menu mei_item = new pf_item_menu ();
        mei_item.setValue ("Bienvenid@ " +
sis_soporte.obtener_instancia_soporte().obtener_variable("nombre_completo"));
        mei_item.addActionListener (crearActionListener ("mbe_index.inicio"));
        mei_item.setUpdate ("dibuja,mensajes");
        mei_item.setIcon ("ui-icon-person");
        mei_item.setOnclick ("var na = $(window).height ();var me =
$('#formulario\\\\:menu').height();var alto = na - me - 45; alto =
parseInt(alto);document.getElementById('formulario:ith_alto').value=alto;");
        mei_item.setOnstart ("document.body.style.cursor = 'wait'");
        mei_item.setOncomplete ("document.body.style.cursor = 'default'");
        this.getChildren ().add (mei_item);

        List lis_consulta = icon_conexion.consultar ("SELECT
tbl_opcion.id_opcion,tbl_opcion.nombre "
                + "FROM tbl_opcion,tbl_perfil_opcion"
                + "WHERE tbl_opcion.id_opcion=tbl_perfil_opcion.id_opcion"
                 + "AND tbl_perfil_opcion.id_perfil=" +      sis_soporte.obtener_instancia_soporte
                 ().obtener_variable ("id_perfil") + " "
                + "AND tbl_opcion.id_opcion_2 IS NULL"
                + "ORDER BY tbl_opcion.nombre");

        for (int i = 0; i < lis_consulta.size (); i++) {
           Object [] fila = (Object []) lis_consulta.get (i);
           Submenu sub_menu = new Submenu ();
           sub_menu.setLabel (fila [1] + "");
           this.getChildren ().add (sub_menu);
          formar_menu_recursivo (sub_menu, fila[0]);
        }
}
```

2.2.3.3 Clase pf_arbol.

Es una implementación del componente Tree de PrimeFaces, permite diseñar un menú tipo árbol lo cual facilita al usuario la elección de opciones, en nuestro proyecto el usuario selecciona los campus de la sede Quito de la Universidad Politécnica Salesiana.

Método configurar_arbol ()

Este método permite definir la tabla de la cual se obtiene información para ser mostrada al usuario, este árbol funciona de forma dinámica dependiendo de la opción que el cliente seleccione en el menú principal.

```
public void configurar_arbol (String tabla, String campo_primaria, String campo_nombre, String
campo_padre, String h) {

        this.tabla = tabla;
        this.campo_primaria = campo_primaria;
        this.campo_nombre = campo_nombre;
        this.campo_padre = campo_padre;
        Object [] raiz = {
           null, h
        };
nodo_raiz = new DefaultTreeNode (raiz, nodo);
nodo_raiz.setExpanded (true);
}
```

2.2.4 Paquete: paq_gestion_academica.

Este paquete contiene ocho clases, estas clases permiten estructurar los módulos de Gestión académica de carreras y el de Disponibilidad de docentes.

2.2.4.1 Gestión de carreras.

La interfaz gráfica de este módulo se maneja de manera simple pero muy funcional, el usuario puede visualizar, modificar, eliminar, insertar u obtener un reporte de la información que se muestra en la pantalla.

Al ingresar a esta pantalla el usuario visualiza en la parte superior un mensaje de bienvenida personalizado con el nombre completo del usuario que inicio sesión, también tiene un menú desplegable denominado Administración, dentro del menú se tiene las opciones para administrar la información académica de la sede Quito como son: sedes, carreras, materias, mallas, niveles, grupos de estudio (matutino, nocturno) y materias paracadémicas.

Para la administración de carreras se definió un menú tipo arbol al lado izquierdo de la pantalla en el cual se muestra la sede y los campus que pertenecen a dicha sede, para esto se crea un objteo del tipo de pf_arbol y se define los paramétros que permiten obtener la información de la sede y campus, además se define la acción a realizar cuando el usuario seleccione una opción en el menú tipo arbol.

```
arb_arbol.setId ("arb_arbol");
arb_arbol.configurar_arbol ("tbl_estructura", "id_estructura", "nombre",
"id_padre","ADMINISTRACION DE CARRERAS");
arb_arbol.onSelect ("seleccionar_arbol","tab_carrera1");
arb_arbol.dibujar ();
```

Para mostrar la información de las carreras se usa clase pf_tabla creando un objeto del tipo pf_tabla, aquí se define la tabla y la forma como la información se mostrará, en este caso se define que el campo id_carrera no sea mostrado y que el campo id_director muestre los nombres de los directores y no su identificador.

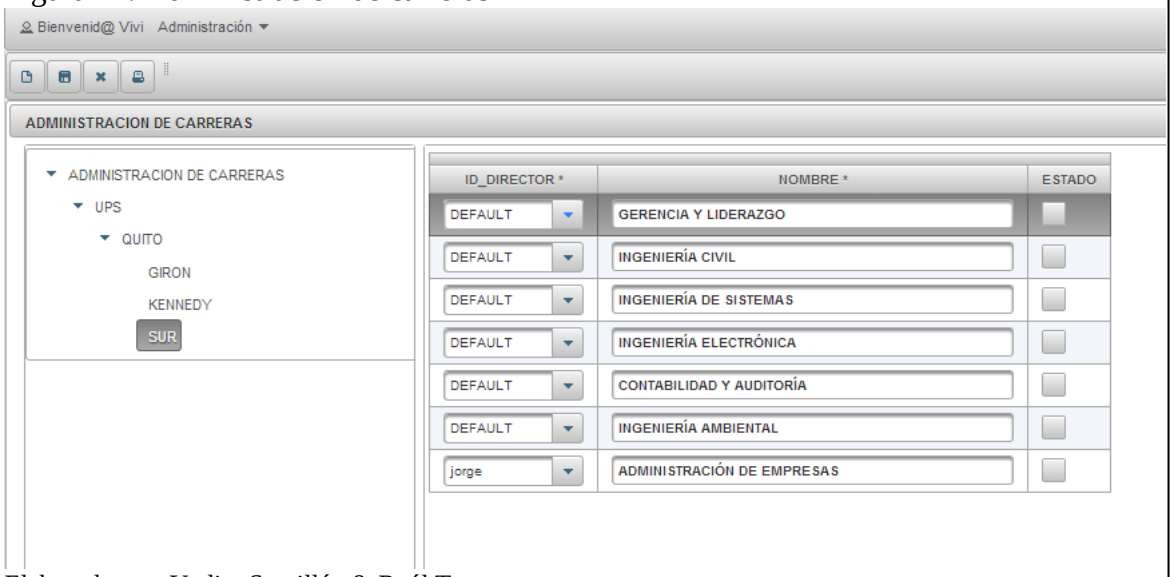
```
tab_carrera.setId ("tab_carrera1");
tab_carrera.setTabla ("tbl_carrera", "id_carrera");
tab_carrera.getColumna ("id_carrera").setVisible (false);
tab_carrera.getColumna ("id_director").configurar_combo ("tbl_usuario", "id_usuario",
"nombre","");
tab_carrera.setCampoPadre ("id_campus");
tab_carrera.dibujar ();
```

Luego se usa la clase pf_panel para definir la posición en la que se mostrarán los componentes usados.

Para esto se define un nombre para identificar al objeto pf_panel y se usa el método dividir2, este recibe cuatro parámetros, el primer y segundo parámetro son los componentes usados, el tercer parámetro es el tamaño que va a ocupar el primer componente y el cuarto parámetro es la forma en la que se muestran los componentes, en este caso se muestran de forma vertical

```
div_carrera.setId ("div_carrera");
div_carrera.dividir2 (arb_arbol, tab_carrera, "22%", "V");
```

Figura 14. Administración de carreras



Elaborado por: Yadira Santillán & Raúl Torres

2.2.4.2 Gestión de materias.

Esta pantalla permite la administración de la información relacionada con las materias, consta de tres paneles separados en forma vertical, en el primer panel se tiene el menú tipo árbol, en el segundo panel se tiene una tabla en modo lectura que contiene la información de las carreras y en el tercer panel se tiene la información de las materias en función de las opciones seleccionadas anteriormente en los dos paneles. La información del tercer panel que contiene la información de materias permite agregar, eliminar o actualizar la información de materias.

Para desplegar la información de los campus se usa el siguiente código en el que se observa el método onSelect (), este método se ejecuta al dar clic sobre uno de los nodos del menú tipo arbol, el valor que se recupera de esta acción al seleccionar en el menú tipo arbol es usado para mostrar la información de carreras.

```
arb_arbol.setId ("arb_arbol");
arb_arbol.configurar_arbol ("tbl_estructura", "id_estructura", "nombre",
"id_padre","ADMINISTRACION DE MATERIAS");
arb_arbol.onSelect ("seleccionar_arbol","tab_tabla_pri");
arb_arbol.dibujar ();
```

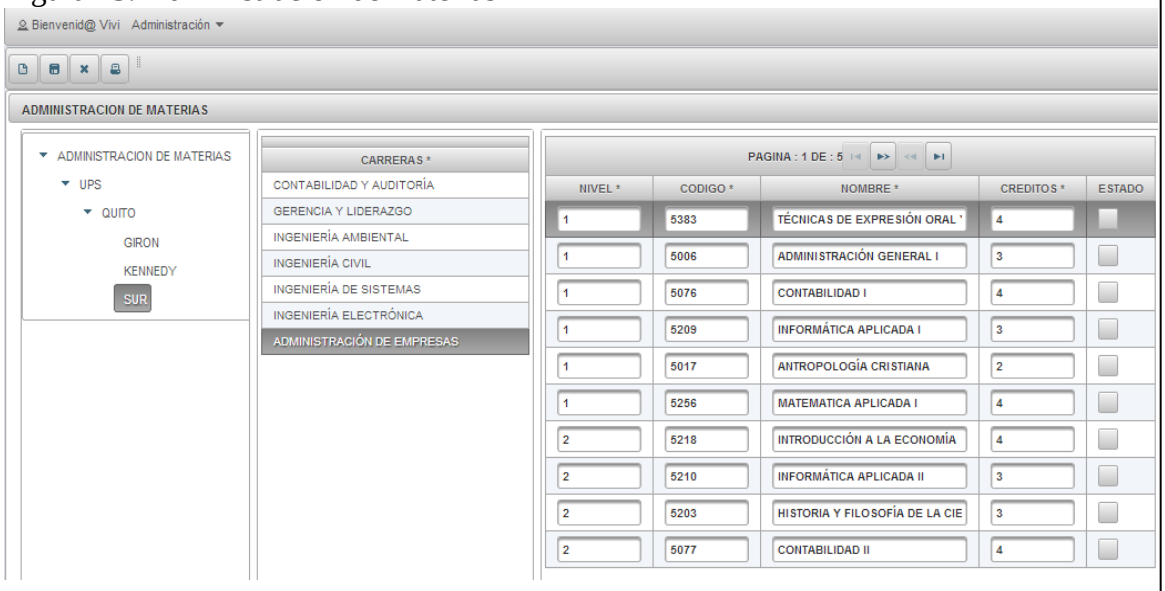
Para mostrar la información de las carreras existentes en el campus seleccionado se usa el siguiente código aquí se tiene el método onSelect (), este método se ejecuta al dar clic sobre los registros de carreras y devuelve el valor de la carrera seleccionada la cual se usa para mostrar las materias que correspondan a la carrera seleccionada.

```
tab_tabla_pri.setId ("tab_tabla_pri");
tab_tabla_pri.setTabla ("tbl_carrera", "id_carrera");
tab_tabla_pri.setCampoPadre ("id_campus");
tab_tabla_pri.setCampoOrden ("id_carrera");
tab_tabla_pri.getColumna ("nombre").setNombre_visual ("CARRERAS");
tab_tabla_pri.setLectura (true);
tab_tabla_pri.getColumna ("id_carrera").setVisible (false);
tab_tabla_pri.getColumna ("id_director").setVisible (false);
tab_tabla_pri.onSelect ("seleccionar_tabla_pri", "tab_tabla_sec");
tab_tabla_pri.agregarRelacion (tab_tabla_sec);
tab_tabla_pri.dibujar ();
pf_panel_tabla pat_panel_pri = new pf_panel_tabla ();
pat_panel_pri.setPanelTabla (tab_tabla_pri);
```

Para mostrar la información de las materias se define la tabla sobre la cual se hará la consulta tomando como parámetro de condición la carrera seleccionada en la segunda tabla (id_carrera)

```
tab_tabla_sec.setId ("tab_tabla_sec");
tab_tabla_sec.setTabla ("tbl_materia", "id_materia");
tab_tabla_sec.setCampoForanea ("id_carrera");
tab_tabla_sec.setCampoOrden ("id_nivel");
tab_tabla_sec.setRows (10);
tab_tabla_sec.getColumna ("id_materia").setVisible (false);
tab_tabla_sec.getColumna ("id_tipo_aula").setVisible (false);
tab_tabla_sec.getColumna ("id_paracademico").setVisible (false);
tab_tabla_sec.getColumna ("id_nivel").setNombre_visual ("NIVEL");
tab_tabla_sec.getColumna ("id_malla").setVisible (false);
tab_tabla_sec.getColumna ("id_malla").setNombre_visual ("NOMBRE-MALLA");
tab_tabla_sec.dibujar ();
pf_panel_tabla pat_panel_sec = new pf_panel_tabla ();
pat_panel_sec.setPanelTabla (tab_tabla_sec);
```

Figura 15. Administración de materias



Elaborado por: Yadira Santillán & Raúl Torres

2.2.4.3 Gestión de mallas.

Esta pantalla administra la información de la malla que está asignada a las carreras existentes en la sede Quito, aquí se puede agregar, actualizar y eliminar una malla, la

información se muestra en una tabla la cual contiene el nombre de la carrera, la malla y el estado de la malla, sí está vigente o no.

Figura 16. Administración de mallas

Elaborado por: Yadira Santillán & Raúl Torres

Para la administración de la información de mallas se usa el siguiente código:

```
tab_malla.setId ("tab_malla");
tab_malla.setTabla ("tbl_malla", "id_malla");
tab_malla.getColumna ("id_carrera").configurar_combo ("tbl_carrera", "id_carrera", "nombre", "");
tab_malla.setCampoOrden ("id_malla");
tab_malla.setRows (6);
tab_malla.dibujar ();
```

2.2.4.4 Gestión de niveles.

La gestión de niveles permite administrar la información relacionada a los niveles que están almacenados en la base de datos, se puede ingresar, actualizar o eliminar un nivel.

El código usado es el siguiente:

```
tbl_nivel.setId ("tbl_nivel");
tbl_nivel.setTabla ("tbl_nivel", "id_nivel");
tbl_nivel.setCondicion ("id_nivel>0");
tbl_nivel.getColumna ("id_nivel").setNombre_visual ("ID");
tbl_nivel.getColumna ("nombre").setNombre_visual ("NIVEL");
tbl_nivel.getColumna ("numero").setVisible (false);
```

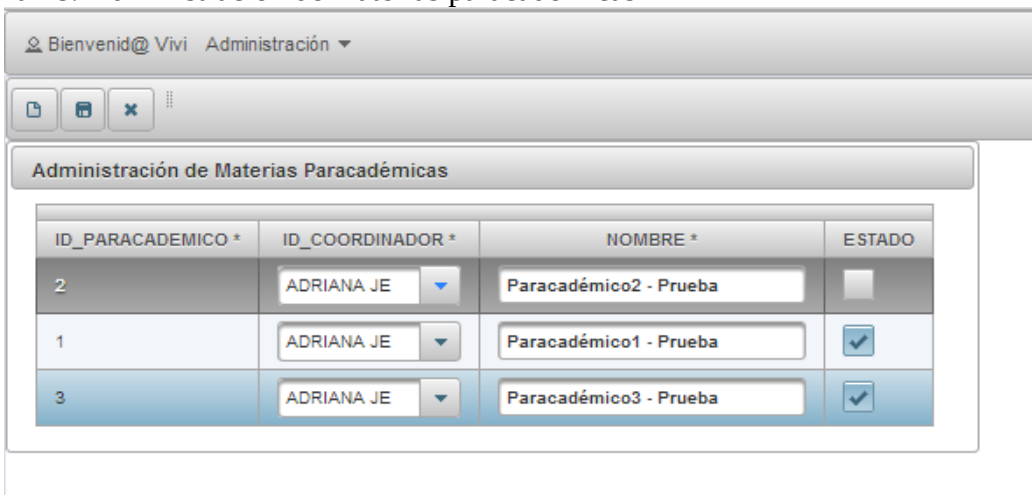
Figura 17. Administración de niveles

Elaborado por: Yadira Santillán & Raúl Torres

2.2.4.5 Gestión de materias paracadémicas.

En esta pantalla se ingresa, modifica o elimina materias paracadémicas, la información de materias paracadémicas tiene el campo, cuando este campo esta desactivado la materia no va a mostrarse en la información de la tabla materias.

Figura 18. Administración de materias paracadémicas



Elaborado por: Yadira Santillán & Raúl Torres

Para representar la información de materias paracadémicas se usa el siguiente código:

```
tbl_paracademico.setId ("tbl_paracademico");
tbl_paracademico.setTabla ("tbl_paracademico", "id_paracademico");
tbl_paracademico.getColumna ("id_coordinador").configurar_combo ("tbl_usuario","id_usuario",
"nombre", "");
tbl_paracademico.setRows (5);
tbl_paracademico.setFocus ();
```

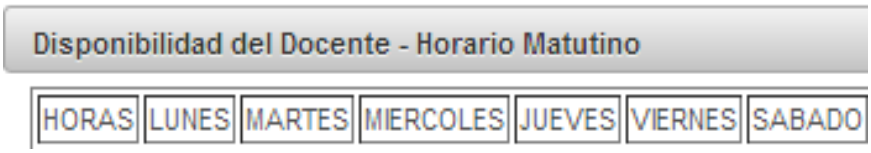
2.2.4.6 Disponibilidad de docentes.

Este módulo permite al usuario ingresar la disponibilidad docente para el horario nocturno y matutino, el usuario selecciona en el menú de disponibilidad el tipo de disponibilidad que desea ingresar. Para el ingreso de la disponibilidad horaria se definió una tabla en la cual se muestra los días de la semana con sus respectivas horas, con cajas de selección para poder elegir el día y hora específica disponible.

Para dibujar en pantalla los días de la semana se consulta a la base de datos para obtener los días y se agrega un bucle for para dibujar la información consultada, para esto se usa el siguiente código.

```
lista_dia = icon_conexion.consultar ("select dia from tbl_dia where id_dia>0 and id_dia<=6 order by
id_dia");
int cont=1;
//se carga los días de la semana con el identificador de las horas
for (int d=0; d<lista_dia.size (); d++){
        HtmlOutputLabel lbldia = new HtmlOutputLabel ();
        lbldia.setId ("lbldia"+d);
        lbldia.setValue (lista_dia.get (d).toString ().toUpperCase());
        lbllist_noc.add (d, lbldia);
```

Figura 19. Cuadrícula - días de la semana



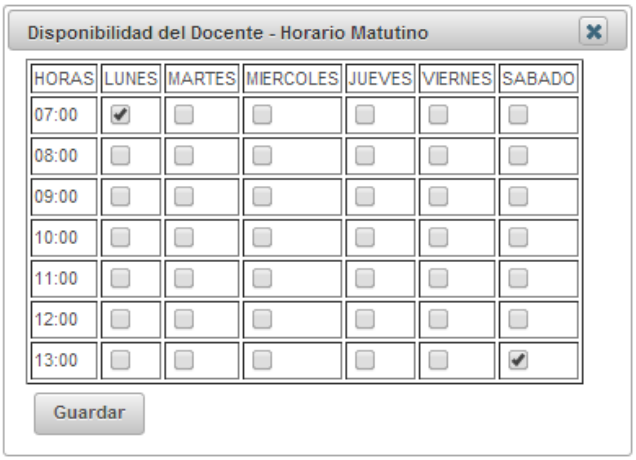
Elaborado por: Yadira Santillán & Raúl Torres

Para la presentación de las horas se hace la consulta a la tabla de horas para obtenere la información de las horas, luego se hace un bucle for para recorrer la lista de la consulta obtenida y se procede a dibujar en pantalla usando cajas de texto de la siguiente manera.

```
if (lista_disp.isEmpty ()) {
            For (int i=0; i<24; i++) {
                        lblhnoc = new HtmlOutputLabel ();
                        rt = new HtmlSelectBooleanCheckbox ();
            if (i==0) {
                        lblhnoc.setId ("lblhnoc5");
                        lblhnoc.setValue (lista_hora.get (0).toString () + ":30");
                        grid_sac_noc.getChildren ().add (lblhnoc);
            } else {
            if (i==6) {
                        lblhnoc.setId ("lblhnoc6");
                        lblhnoc.setValue (lista_hora.get (1).toString () + ":30");
                        grid_sac_noc.getChildren ().add (lblhnoc);
            } else {
            if (i==12) {
                        lblhnoc.setId ("lblhnoc7");
                        lblhnoc.setValue (lista_hora.get (2).toString () + ":30");
                        grid_sac_noc.getChildren ().add (lblhnoc);
            } else {
            if (i==18) {
                        lblhnoc.setId ("lblhnoc8");
                        lblhnoc.setValue (lista_hora.get (3).toString () + ":30");
                        grid_sac_noc.getChildren ().add (lblhnoc);
            } else {
            if (i==24){
                        lblhnoc.setId ("lblhnoc9");
                        lblhnoc.setValue (lista_hora.get (4).toString () + ":30");
                        grid_sac_noc.getChildren ().add (lblhnoc);
            }}}
}
          if((i!=0)||(i!=6)||(i!=12)||(i!=18)||(i!=24)){
                    rt.setId ("rt"+i);
                    chk_noc.add (i, rt);
                    grid_sac_noc.getChildren ().add (chk_noc.get (i));
          }}
```

Como resultado se obtiene una cuadrícula que contiene los días y horas de trabajo con cajas de selección y un botón que permite guardar la información sobre la disponibilidad de docentes.

Figura 20. Cuadrícula – horas de la semana



Elaborado por: Yadira Santillán & Raúl Torres

Cuando el docente que ingresa al sistema no tiene registrado una disponibilidad horaria anterior se genera la cuadrícula de disponibilidad vacía y cuando el docente tiene disponibilidad horaria almacenada se muestra una cuadrícula con los datos del semestre anterior, de esta forma el docente tiene una idea clara de los días trabajados el semestre anterior y si desea puede modificar la información y guardarla para el nuevo semestre.

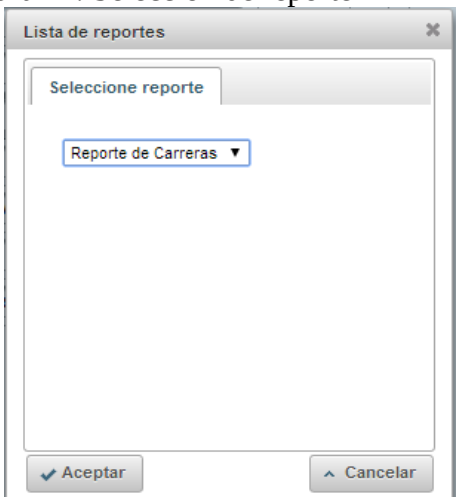
2.2.4.7 Reportes.

Para el módulo de Gestión académica de carreras se ha definido obtener reportes de carreras, materias, materias paracadémicas, mallas y disponibilidad de docentes se ha definido tres formatos para la generación de reportes que son PDF, Word y Excel.

Para obtener el reporte de carreras, materias, materias paracadémicas o mallas se debe elegir en el menú de Administración la opción que deseemos, realizar las búsqueda de la información para que se cargue en las tablas, luego hacer clic en el icono de reporte que se encuentra en la parte superior izquierda de la pantalla.

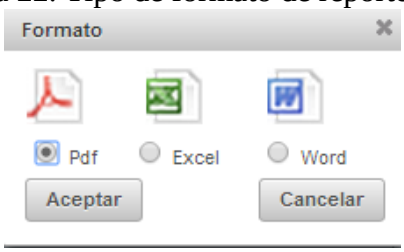
Al dar clic en reporte se despliega una ventana emergente en la cual se selecciona el tipo de formato (PDF, CSV, DOC) con el que se requiere generar el reporte.

Figura 21. Selección de reporte



Elaborado por: Yadira Santillán & Raúl Torres

Figura 22. Tipo de formato de reporte



Elaborado por: Yadira Santillán & Raúl Torres

Como resultado se genera un archivo con información de las carreras, materias, materias paracadémicas o mallas según la opción seleccionada.

Figura 23. Reporte en formato PDF

REPORTE DE CARRERAS

UPS Schedule

ID	CARRERA
4	CONTABILIDAD Y AUDITORÍA
7	GERENCIA Y LIDERAZGO
9	INGENIERÍA AMBIENTAL
10	INGENIERÍA CIVIL
11	INGENIERÍA DE SISTEMAS
13	INGENIERÍA ELECTRÓNICA
18	ADMINISTRACIÓN DE EMPRESAS

Elaborado por: Yadira Santillán & Raúl Torres

Para el diseño de los reportes se hace uso del entorno gráfico Ireport el cual permite diseñar el reporte y para compilar y ejecutar el reporte se usa las librerías commons-

beanutils-1.8.0.jar, commons-collections-3.2.1.jar, commons-digester-1.7.jar, commons-Javaflow-20060411.jar, commons-loggin-1.1.jar, jasperreports-4.5.0.jar y en especial itext-2.1.7.jar la que permite generar nuestros reportes en formato PDF y en otros formatos como los usados en el proyecto (Word, Excel).

2.3 Pruebas

Las pruebas se realizaron por cada módulo del sistema, se toma en cuenta la funcionalidad, rapidez, interfaz gráfica de usuario, acceso al sistema, despliegue e interacción con la información almacenada en la base de datos, las novedades encontradas se describen a continuación:

2.3.1 Pruebas contra requerimientos.

Se realiza las pruebas contra requerimientos con cada uno de los usuarios del sistema para verificar sus requerimientos funcionales y específicos.

2.3.1.1 Pruebas funcionales.

En las pruebas funcionales se confirma que el proyecto ya finalizado cumpla con los requerimientos funcionales y específicos y son representadas con la tabla 27 para el usuario Administrador y docente en la tabla 28.

Administrador

Tabla 32. Prueba funcional – Administrador

N°	Nombre de la Prueba	Req.	Pasos	OK	Observaciones
1	Crear carreras	1.1	1.- Acceder a la página de administración 2.- Seleccionar la opción de carreras 3.- Seleccionar la sede 4.- Clic en el ícono Insertar (opcional clic derecho - Insertar) 5.- Llenar los campos obligatorios 6.- Clic en el ícono Guardar (opcional clic derecho - Guardar)	Sí	Se puede crear varios registros en la misma transacción y luego guardarla

Continúa …

Tabla 32. Prueba funcional – Administrador (Continuación…)

N°	Nombre de la Prueba	Req.	Pasos	OK	Observaciones
2	Modificar carreras	1.1	1.- Acceder a la página de administración 2.- Seleccionar opción de carreras 3.- Seleccionar la sede 4.- Modificar los datos 5.- Clic en el ícono Guardar (clic derecho - Guardar)	Sí	Se puede modificar varios registros en una sola transacción antes de guardarla
3	Eliminar carreras	1.1	1.- Acceder a la página de administración 2.- Seleccionar la opción de carreras 3.- Seleccionar la sede 4.- Clic en el ícono Eliminar (clic derecho - Eliminar) 5.- Clic en el ícono Guardar (clic derecho - Guardar)	Sí	Se puede eliminar varios registros en una sola transacción antes de guardarla
4	Crear Materias	1.2	1.- Acceder a la página de administración 2.- Seleccionar la opción de Materias 3.- Seleccionar la sede 4.- Seleccionamos la Carrera 5.- Clic en el ícono Insertar (opcional clic derecho - Insertar) 6.- Llenar los campos obligatorios 7.- Clic en el ícono Guardar (opcional clic derecho - Guardar)	Sí	Se puede crear varios registros en la misma transacción y luego guardarla
5	Modificar Materias	1.2	1.- Acceder a la página de administración 2.- Seleccionar la opción de Materias 3.- Seleccionar la sede 4.- Seleccionar la Carrera 5.- Modificar los datos 6.- Clic en el ícono Guardar (clic derecho - Guardar)	Sí	Se puede modificar varios registros en una sola transacción antes de guardarla

Continúa…

Tabla 32. Prueba funcional – Administrador (Continuación…)

N°	Nombre de la Prueba	Req.	Pasos	OK	Observaciones
6	Eliminar Materias	1.2	1.- Acceder a la página de administración 2.- Seleccionar la opción de Materias 3.- Seleccionar la sede 4.- Seleccionar la Carrera 5.- Sobre el registro que vamos a eliminar dar clic en el ícono Eliminar (clic derecho - Eliminar) 6.- Clic en el ícono Guardar (clic derecho - Guardar)	Sí	Se puede eliminar varios registros en una sola transacción antes de guardarla
7	Crear Malla	1.3	1.- Acceder a la página de administración 2.- Seleccionar la opción de Mallas 3.- Clic en el ícono Insertar (opcional clic derecho - Insertar) 4.- Llenar los campos obligatorios 5.- Clic en el ícono Guardar (opcional clic derecho - Guardar)	Sí	Se puede crear varios registros en la misma transacción y luego guardarla
8	Modificar Malla	1.3	1.- Acceder a la página de administración 2.- Seleccionar la opción de Mallas 3.- Modificar los datos 4.- Clic en el ícono Guardar (clic derecho - Guardar)	Sí	Se puede modificar varios registros en una sola transacción antes de guardarla
9	Eliminar Malla	1.3	1.- Acceder a la página de administración 2.- Seleccionar la opción de Mallas 3.- Sobre el registro que vamos a eliminar dar clic en el ícono Eliminar (clic derecho - Eliminar) 4.- Clic en el ícono Guardar (clic derecho - Guardar)	Sí	Se puede eliminar varios registros en una sola transacción antes de guardarla
10	Crear Nivel	1.4	1.- Acceder a la página de administración 2.- Seleccionar la opción de Nivel 3.- Clic en el ícono Insertar (opcional clic derecho - Insertar) 4.- Llenar los campos obligatorios 5.- Clic en el ícono Guardar (opcional clic derecho - Guardar)	Sí	Se puede crear varios registros en la misma transacción y luego guardarla
11	Modificar Nivel	1.4	1.- Acceder a la página de administración 2.- Seleccionar la opción de Nivel 3.- Modificar los datos del registro que se desee 4.- Clic en el ícono Guardar (clic derecho - Guardar)	Sí	Se puede modificar varios registros en una sola transacción antes de guardarla

Continúa…

Tabla 32. Prueba funcional – Administrador (Continuación…)

N°	Nombre de la Prueba	Req.	Pasos	OK	Observaciones
12	Eliminar Nivel	1.4	1.- Acceder a la página de administración 2.- Seleccionar la opción de Nivel 3.- Sobre el registro que vamos a eliminar dar clic en el ícono Eliminar (clic derecho - Eliminar) 4.- Clic en el ícono Guardar (clic derecho - Guardar)	Sí	Se puede eliminar varios registros en una sola transacción antes de guardarla
13	Crear Paracadémico	1.5	1.- Acceder a la página de administración 2.- Seleccionar la opción de Paracadémico 3.- Clic en el ícono Insertar (opcional clic derecho - Insertar) 4.- Llenar los campos obligatorios 5.- Clic en el ícono Guardar (opcional clic derecho - Guardar)	Sí	Se puede crear varios registros en la misma transacción y luego guardarla
14	Modificar Paracadémico	1.5	1.- Acceder a la página de administración 2.- Seleccionar la opción de Paracadémico 3.- Modificar los datos del registro que se desee 4.- Clic en el ícono Guardar (clic derecho - Guardar)	Sí	Se puede modificar varios registros en una sola transacción antes de guardarla
15	Eliminar Paracadémico	1.5	1.- Acceder a la página de administración 2.- Seleccionar la opción de Paracadémico 3.- Sobre el registro que vamos a eliminar dar clic en el ícono Eliminar (clic derecho - Eliminar) 4.- Clic en el ícono Guardar (clic derecho - Guardar)	Sí	Se puede eliminar varios registros en una sola transacción antes de guardarla

Elaborado por: Yadira Santillán & Raúl Torres

Docente

Tabla 33. Prueba funcional - Docente

N°	Nombre de la Prueba	Req.	Pasos	OK	Observaciones
1	Insertar disponibilidad horaria	1.1	1.- Acceder al sistema como docente 2.- Seleccionar la opción Matutino/Nocturno 3.- Activar dando clic en las cajas de validación (check) 4.- Clic en el botón guardar	Sí	Se puede ingresar (activar) varias cajas de validación (check) en la misma transacción y luego guardarla
2	Modificar disponibilidad horaria	1.2	1.- Acceder al sistema como docente 2.- Seleccionar la opción Matutino/Nocturno 3.- Desactivar dando clic en las cajas de validación (check) 4.- Clic en el botón guardar	Sí	Se puede modificar (desactivar) varias cajas de validación (check) en la misma transacción y luego guardarla

Elaborado por Yadira Santillán & Raúl Torres

2.3.2 Pruebas de rendimiento.

Para las pruebas de rendimiento se toma en cuenta el tiempo promedio que el módulo de GAC y DID individualmente tardan al mostrar la interfaz gráfica de usuario y al realizar transacciones sobre cada pantalla de los módulos, para esto se usó como navegadores Google Chrome (GC) y Mozilla Firefox (MF), cabe mencionar que se realizó diez pruebas por cada una de las pantallas.

La medición de tiempo de respuesta de nuestro proyecto se lo hizo usando el complemento firebug para el navegador Firefox y la opción de herramientas del desarrollador en Google Chrome, los tiempos obtenidos son representados en gráficas de barras las que permiten evidenciar de forma clara que navegador trabaja de mejor forma, esto permite definir con cuál de estos dos navegadores trabajar para tener respuestas óptimas al momento usar los módulos de GAC y DID.

Tabla 34. Prueba de rendimiento-funcionalidad de pantallas de GAD y DID

#	NOMBRE DE LA PANTALLA		TIEMPO			OK	OBSERVACIONES
			GOOGLE CHROME (GC)	INTERNET EXPLORER	MOZILLA FIREFOX (MF)		
1	Ingresar al sistema		425 ms	328 ms	447 ms	Sí	
2	Pantalla principal de administración académica		549 ms	359 ms	682 ms	Sí	
3	Pantalla de administración de carreras		66 ms	234 ms	438 ms	Sí	
4	Transacciones sobre la pantalla administración de carreras	Insertar	257 ms	125 ms	170 ms	Sí	Ingresa un solo registro
		Actualizar	218 ms	234 ms	160 ms	Sí	Actualiza un solo registro
		Eliminar	148 ms	156 ms	253 ms	Sí	Elimina un solo registro
5	Reporte de carreras		9,41 s	8,75 s	2,12 s	Sí	Cuando muestra 9 registros
6	Pantalla de administración de materias		252 ms	124 ms	531 ms	Sí	
7	Transacciones sobre la pantalla de administración de materias	Insertar	124 ms	320 ms	424 ms	Sí	Ingresa un solo registro
		Actualizar	30 ms	312 ms	95 ms	Sí	Actualiza un solo registro
		Eliminar	22 ms	90 ms	47 ms	Sí	Elimina un solo registro
8	Reporte de materias		7,09 s	1,48 s	7,46 ms	Sí	Obtiene reporte de 70 registros
9	Pantalla de administración de mallas		56 ms	218 ms	62 ms	Sí	
10	Transacciones sobre la pantalla de administración de mallas	Insertar	47 ms	187 ms	54 ms	Sí	Ingresa un solo registro
		Actualizar	39 ms	109 ms	64 ms	Sí	Actualiza un solo registro
		Eliminar	31 ms	93 ms	48 ms	Sí	Elimina un solo registro
11	Pantalla de administración de Sedes		388 ms	124 ms	69 ms	Sí	
12	Transacciones sobre la pantalla de administración de sedes	Insertar	177 ms	180 ms	48 ms	Sí	Inserta un solo registro
		Actualizar	48 ms	187 ms	31 ms	Sí	Actualiza un solo registro
		Eliminar	181 ms	468 ms	31 ms	Sí	Elimina un solo registro
13	Pantalla de administración de niveles		72 ms	156 ms	49 ms	Sí	
14	Transacciones sobre la pantalla de niveles	Insertar	73 ms	109 ms	32 ms	Sí	Inserta un solo registro
		Actualizar	26 ms	172 ms	28 ms	Sí	Actualiza un solo registro
		Eliminar	22 ms	94 ms	27 ms		
15	Pantalla de administración de materias paracadémicas		356 ms	219 ms	125 ms	Sí	

Continúa…

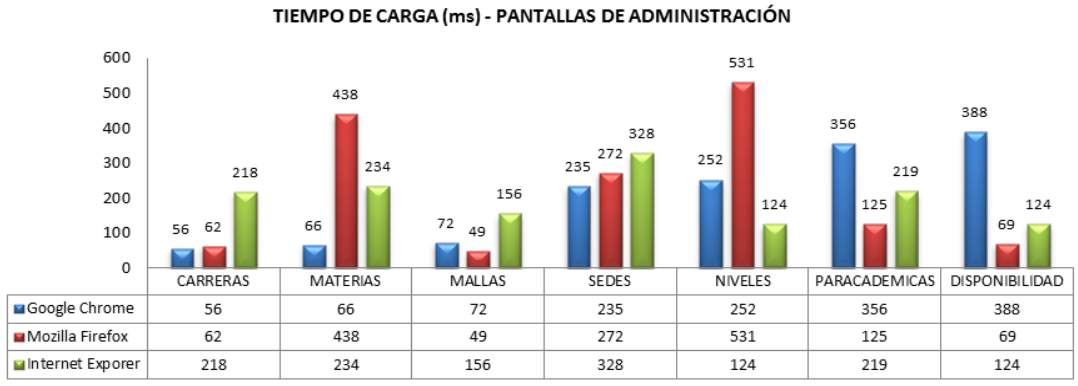
Tabla 34. Prueba de rendimiento-funcionalidad de pantallas de GAD y DID (Continuación…)

#	NOMBRE DE LA PANTALLA		TIEMPO GOOGLE CHROME (GC)	TIEMPO INTERNET EXPLORER	TIEMPO MOZILLA FIREFOX (MF)	OK	OBSERVACIONES
16	Transacciones sobre la pantalla de administración de materias paracadémicas	Insertar	123 ms	125 ms	122 ms	Sí	Inserta un solo registro
		Actualizar	118 ms	126 ms	96 ms	Sí	Actualiza un solo registro
		Eliminar	76 ms	62 ms	63 ms	Sí	Elimina un solo registro
17	Pantalla de Ingreso de Disponibilidad de docentes		235 ms	328 ms	272 ms	Sí	
18	Transacciones sobre la pantalla de disponibilidad de docentes	Insertar	382 ms	359 ms	499 ms	Sí	Inserta los registros que se muestran en la malla de disponibilidad
		Actualizar	322 ms	327 ms	298 ms	Sí	Actualiza los registros que se muestran en la malla de disponibilidad

Elaborado por Yadira Santillán & Raúl Torres

Las siguientes figuras muestran cual fue el desempeño de los navegadores usados en los módulos de GAC y DID, hay que tomar en cuenta que los tiempos de respuesta varían entre los dos navegadores por la forma en la que cada uno trabaja. Para el caso de las pantallas de administración se observa que el navegador Google Chrome tiene un mejor desempeño al tener mejores tiempos de respuesta en mostrar la información y realizar transacciones.

Figura 24. Tiempo de carga (ms) en pantallas de administración

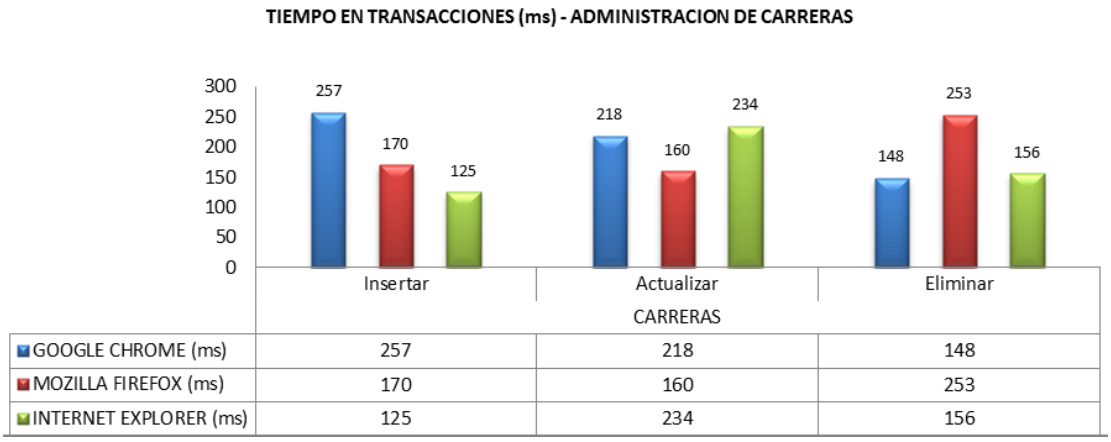


Elaborado por Yadira Santillán & Raúl Torres

El explorador Mozilla Firefox tiende a tardar más tiempo cuando se debe mostrar varios registros en pantalla, esto se identifica en la pantalla de administración de materias al tener un tiempo de respuesta de 438 ms, lo cual es un tiempo alto al comparar con el tiempo de respuesta de 66 ms que se tiene en Google Chrome y 234 en Internet Explorer.

En la medición del tiempo de respuesta de las transacciones que se realizan en la pantalla de administración de carreras se identifica que al insertar y actualizar datos sobre la base el navegador Mozilla Firefox responde en menos tiempo que Google Chrome e Internet Explorer, a excepción cuando se elimina un registro ya que Google Chrome tiene un tiempo de respuesta menor.

Figura 25. Tiempo en transacciones de AC

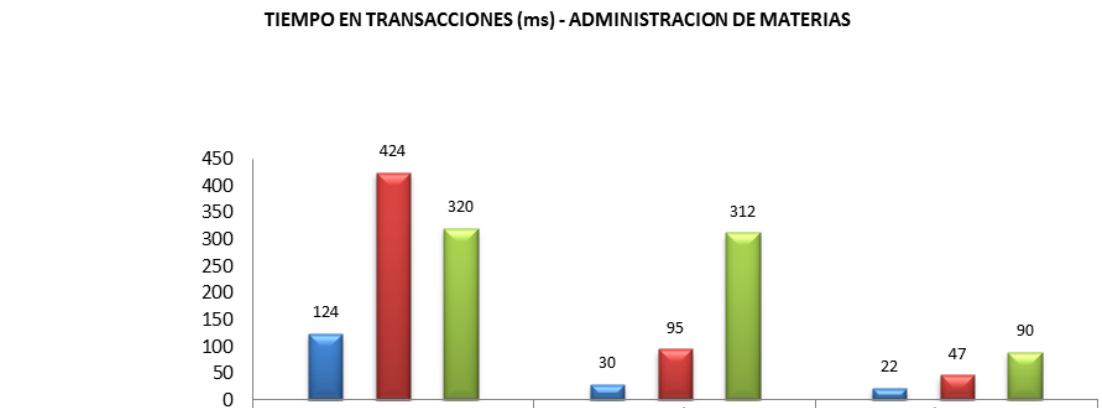


	Insertar	Actualizar	Eliminar
GOOGLE CHROME (ms)	257	218	148
MOZILLA FIREFOX (ms)	170	160	253
INTERNET EXPLORER (ms)	125	234	156

Elaborado por Yadira Santillán & Raúl Torres

En la pantalla de administración de materias se tiene una carga de información más alta en comparación con el resto de pantallas, según las pruebas realizadas se identifica que Google Chrome responde de forma más rápida al interactuar el sistema con la base de datos tomando en cuenta que la información con la que se trabaja en esta pantalla se obtiene de cuatro tablas distintas.

Figura 26. Tiempo en transacciones de administración de materias

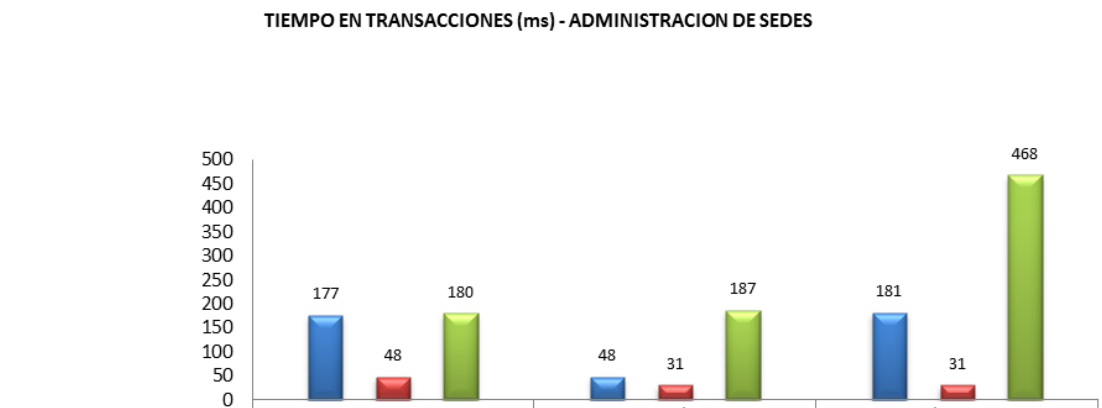


	Insertar	Actualizar	Eliminar
GOOGLE CHROME (ms)	124	30	22
MOZILLA FIREFOX (ms)	424	95	47
INTERNET EXPLORER (ms)	320	312	90

Elaborado por Yadira Santillán & Raúl Torres

En la medición de las transacciones en la pantalla de administración de sedes el navegador Mozilla Firefox tuvo un mejor desempeño al tener respuestas más cortas para realizar insercciones, actualizaciones y eliminaciones en la base de datos.

Figura 27. Tiempo en transacciones de administración de sedes

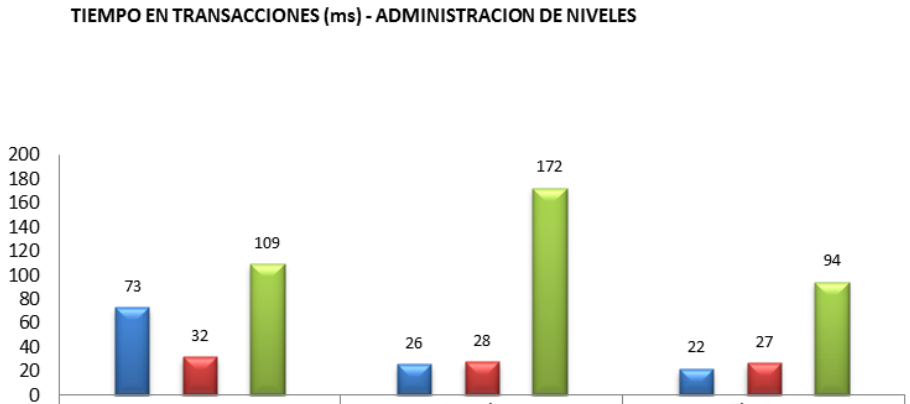


	Insertar	Actualizar	Eliminar
GOOGLE CHROME (ms)	177	48	181
MOZILLA FIREFOX (ms)	48	31	31
INTERNET EXPLORER (ms)	180	187	468

Elaborado por Yadira Santillán & Raúl Torres

En el caso de la pantalla de administración de niveles, materías paracadémicas y mallas, se visualiza que los dos navegadores tienen un comportamiento similar, esto se asume a la poca carga de información que demanda la pantalla de administración de niveles y materias paracadémicas.

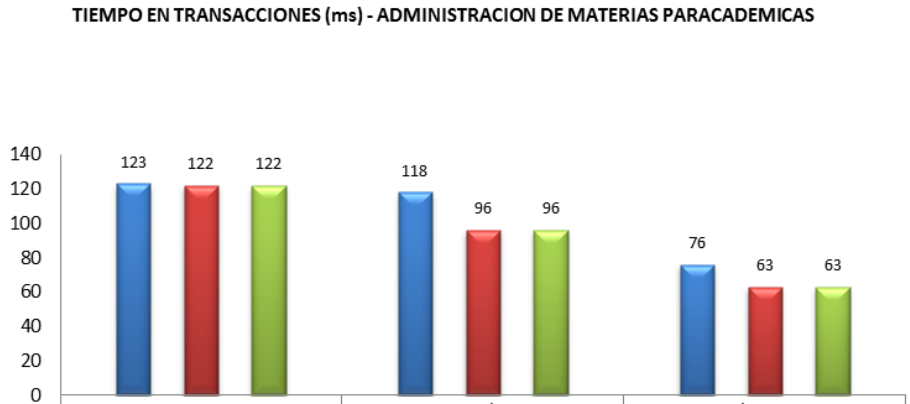
Figura 28. Tiempo en transacciones de administración de niveles



	Insertar	Actualizar	Eliminar
GOOGLE CHROME (ms)	73	26	22
MOZILLA FIREFOX (ms)	32	28	27
INTERNET EXPLORER (ms)	109	172	94

Elaborado por Yadira Santillán & Raúl Torres

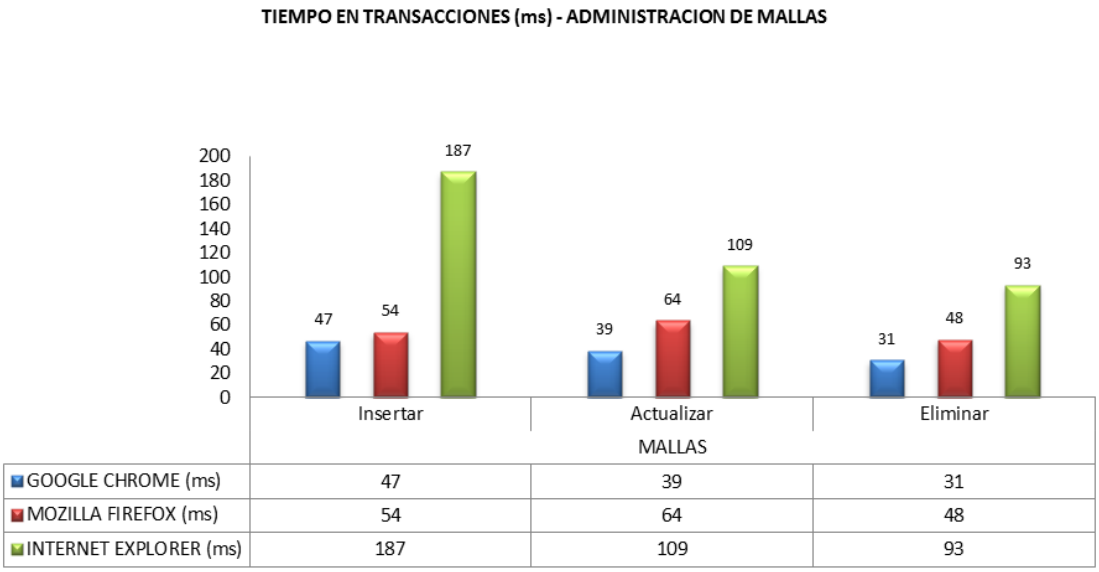
Figura 29. Tiempo en transacciones de administración de materias paracadémicas



	Insertar	Actualizar	Eliminar
GOOGLE CHROME (ms)	123	118	76
MOZILLA FIREFOX (ms)	122	96	63
INTERNET EXPLORER (ms)	122	96	63

Elaborado por Yadira Santillán & Raúl Torres

Figura 30. Tiempo en transacciones de administración de mallas

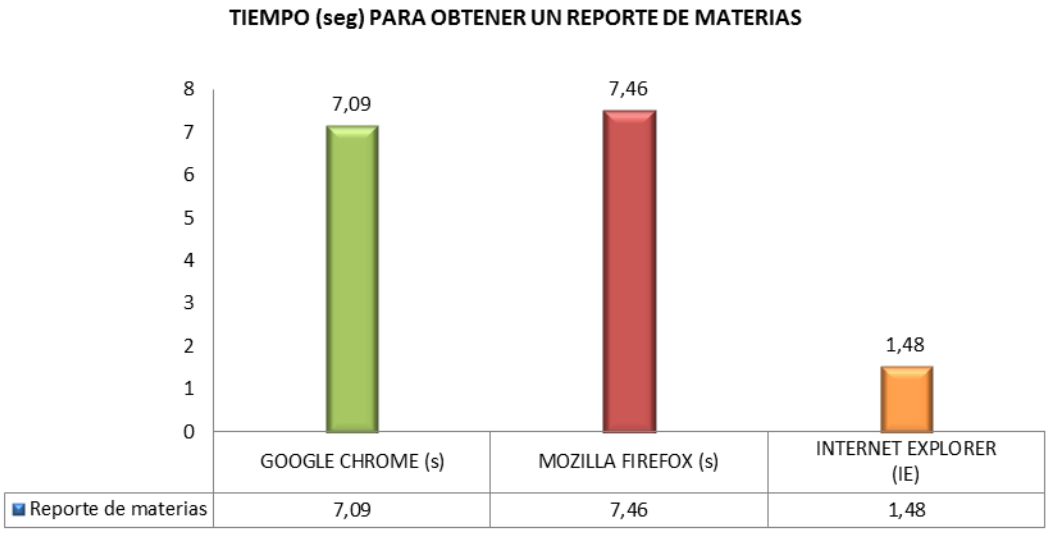


	Insertar	Actualizar	Eliminar
GOOGLE CHROME (ms)	47	39	31
MOZILLA FIREFOX (ms)	54	64	48
INTERNET EXPLORER (ms)	187	109	93

Elaborado por Yadira Santillán & Raúl Torres

Cuando el usuario genera un reporte se identifica que el navegador Google Chrome tarda más tiempo que Mozilla Firefox en descargar el reporte ya sea el de carreras o materias, Internet Explorer

Figura 31. Tiempo en transacciones al generar reporte de materias



	GOOGLE CHROME (s)	MOZILLA FIREFOX (s)	INTERNET EXPLORER (IE)
Reporte de materias	7,09	7,46	1,48

Elaborado por Yadira Santillán & Raúl Torres

CAPÍTULO 3

INTEGRACIÓN

El sistema UPS Schedule, está conformado por módulos desarrollados por distintos equipos de trabajo, cada uno de estos equipos debe definir y realizar las adaptaciones necesarias, a la estructura de la base de datos y clases, obteniendo un funcionamiento óptimo de cada módulo; éstas adaptaciones, generan diferencias que se requieren solventar para la unificación de los módulos y obtener el funcionamiento correcto del sistema integrado.

Para facilitar la integración de los módulos, al inicio del desarrollo del sistema, se realizó un diseño de la estructura de la base de datos, sobre el cual cada uno de los equipos de trabajo, puede desarrollar las modificaciones según los requerimientos de cada módulo.

Para la integración de los módulos, se realizó un análisis e intercambio de los datos que fueron modificados sobre la estructura de la base de datos inicial, cada uno de los equipos de trabajo generó cambios, que fueron considerados para la estructura final de la base de datos, en este proceso se conserva la funcionalidad para cada uno de los módulos y de todo el sistema evitando causar conflicto entre ellos.

En esta etapa del proyecto se integran las tablas y los campos utilizados en los diferentes módulos, para ello se realizó el siguiente proceso:

- Se intercambió datos sobre las modificaciones que cada uno de los módulos realizó sobre la estructura inicial del diseño de la base de datos.
- Se verificó cada uno de los nombres de las tablas, definiendo un nombre común para cada una de ellas.
- Se analizó y definió el tipo de variable para los campos de las tablas.
- Se analizó y acordó añadir campos en determinadas tablas, que se requerían.

- Se optimizó los campos considerando el costo futuro de las consultas, utilizando el campo id_carrera como referencia en varias tablas.
- Se realizó la unión de todas las tablas con sus respectivas relaciones, formando una sola base de datos, que permita el funcionamiento del sistema UPS Schedule.

Al realizar la integración de la base de datos, se analiza e integra los paquetes que contienen las clases desarrolladas o modificadas por cada módulo, de esta manera se obtienen las clases finales, que a su vez forman parte del sistema.

Para la verificación del proceso de integración se realizaron las siguientes actividades.

- Análisis e integración de variables y métodos
- Compilar y ejecutar el proyecto.
- Ingresar al sistema UPS Schedule con el usuario administrador.
- Usar el módulo permisos que corresponde a la clase sis_permiso, este otorga permisos de ejecución a los paquetes y clases de cada grupo de la siguiente forma:
    - Se asigna un nombre al paquete permitirá ejecutarse
    - Se ingresa el nombre original del paquete
    - Para las clases de igual forma se asigna un nombre
    - Se ingresa el nombre original de la clase
    - Se ingresa el nombre del paquete al que pertenece

De esta forma, se prueba la adecuada integración de las distintas clases, trabajadas por cada módulo permitiendo realizar pruebas que impliquen la utilización de las validaciones integradas en los distintos módulos, como es el caso de la prueba realizada dónde:

Se verifica que el administrador del sistema se autentifica, ingresa al sistema con su respectivo perfil, ingresa al módulo integrado de administración de permisos, donde se otorga los permisos que sean necesarios a las clases que se considere para modificarlas, y lleva un control de los cambios realizados, por medio de una auditoría sobre cualquier acción que realicen las clases que fueron añadidas.

Con el fin de evitar problemas de compatibilidad en el desarrollo e integración de los distintos módulos, cada módulo fue desarrollado en la herramienta de desarrollo de lenguaje Java/JSF con el IDE NetBeans la versión 7.1.2, la librería para reportes Ireport la versión 4.5.0 y para la edición Jasper versión 4.5.0.

CONCLUSIONES

- El análisis realizado para el diseño de los módulos GAC y DID permitió determinar la necesidad de establecer variables de entrada y salida, que faciliten el desarrollo e integración con los demás módulos, siendo estos id de usuario, disponibilidad de docentes y gestión de las carreras.

- Se utilizó el framework X-PRIME que se encuentra desarrollado en lenguaje JSF, este framework es una adaptación de PrimeFaces, facilitando así una variedad de componentes los mismos que ayudaron a la creación del diseño de la interfaz gráfica de usuario de los distintos módulos.

- Al realizar las pruebas de rendimiento con el componente Firebug de Mozilla Firefox y las herramientas del desarrollador de Google Chrome e Internet Explorer, se obtuvo que el mejor tiempo promedio de respuesta es de 123ms, concluyendo así que el navegador Google Chrome es más rápido al realizar las transacciones sobre los módulos GAC y DID.

- Al realizar la integración de los módulos fue necesario el análisis e integración de variables y métodos, el diseño y la estructura de la base de datos que fue posible gracias a un trabajo colaborativo entre los tesistas abriendo la posibilidad de que exista un intercambio de conocimientos, ideas, debates con planteamiento de soluciones, aprendizaje y uso de herramientas colaborativas como Dropbox y SourceForge.

RECOMENDACIONES

- Para una mejor estructura de la base de datos se debe previamente definir el tipo de variable para cada una de las tablas, así se evita que en el proceso de integración aparezcan diferencias en la información almacenada en la base de datos.

- Para futuros trabajos se recomienda considerar la disponibilidad de horas administrativas, investigación etc. Para que el director de carrera tenga una mejor apreciación del tiempo que sus docentes pueden dedicar a la universidad.

- El proyecto actual podría ser repotenciado al desarrollar una aplicación móvil la que permita acceder a cualquier hora y desde cualquier lugar al usuario permitiéndole una independencia de un equipo de computación fijo, esta mejora puede ser desarrollada también usando librerías primefaces.

- Para ejecutar las pruebas de rendimiento se debe considerar siempre dos o tres exploradores para definir cuál es el explorador que trabaja de forma rápida de forma que se pueda recomendar al usuario final su uso para un mejor desempeño de los módulos GAC y DID.

LISTA DE REFERENCIAS

Cabello, J. C. (2010). *Diseño de paginas web con XHTML, Javascript y CSS, 3ra edición.* RA-MA S.A.

Douglas, K. D.-S. (2003). *PostgreSQL - A comprehensive guide to building programming and administering PostgresQL database.* Sams Publishing.

Murphy, P. T. (2010). *Persistence with Hibernate.* Estados Unidos.

Mann, K. D. (2005). *Java Server Faces in action.* Estados Unidos: Manning.

Raghu Kodali. Jonathan Wetherbee, P. Z. (2006). *Beginning EJB 3 Aplication Develpoment.*

Schmuller, J. (2003). *Aprendiendo UML en 24 horas. 1ra. Edición.* México DF: Pretince Hall.

Çivici, Ç. (2012). primefaces_users_guide. En Ç. Çivici, *primefaces_users_guide* (pág. 475).

Almirón, C. G. (26 de marzo del 2009). *Introducción a JSF Java*. Recuperado el 4 de enero del 2014, de http://www.adictosaltrabajo.com/tutoriales/tutoriales.php?pagina=IntroduccionJSFJava#_Toc225422690

Corporation, O. (19 de noviembre del 2013). *Bienvenido a NetBeans y www.netbeans.org.* Recuperado el 23 de diciembre del 2013, de https://netbeans.org/index_es.html

Cruz, G. F. (30 de agosto del 2012). *slideshare.* Recuperado el 23 de diciembre de 2013, de http://www.slideshare.net/gus_farfan/primefaces-14115155

Guevara, J. M. (10 de octubre de 2009). *Fundamentos de programación en Java.* Recuperado el 23 de diciembre del 2013, de http://pendientedemigracion.ucm.es/info/tecnomovil/documentos/fjava.pdf

Manchado, D. S. (junio del 2010). *Estudio del servidor de aplicaciones Glassfish y de las aplicaciones J2EE.* Recuperado el 23 de diciembre del 2013, de http://ddd.uab.cat/pub/trerecpro/2013/hdl_2072_206748/SerraManchadoDavidR-ETISa2009-10.pdf

Mora, A. (02 de diciembre de 2011). *Diagrama de bloques*. Recuperado el 08 de enero del 2014, de http://www.slideshare.net/AdrianaMora1/diagrama-en-bloques

Torrijos, R. L. (s.f.). *Introducción a la Tecnología JavaServer Faces.* Recuperado el 23 de diciembre de 2013, de http://www.programacion.com/articulo/introduccion_a_la_tecnologia_javaserver_faces_233